THE UNIVERSITY OF ALBERTA

RELEASE FORM

NAME OF AUTHOR RAKESH K. JAIN

TITLE OF THESIS "COMPUTER MATCHING OF STEREO PICTURES"

DEGREE FOR WHICH THESIS WAS PRESENTED M.Sc.

YEAR THIS DEGREE GRANTED 1977

Permission is hereby granted to THE UNIVERSITY OF ALBERTA LIBRARY to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly or scientific research purposes only.

The author reserves other publication rights, and neither the thesis nor extensive extracts from it may be printed or otherwise reproduced without the author's written permission.

THE UNIVERSITY OF ALBERTA

COMPUTER MATCHING OF STEREO PICTURES

by

© RAKESH K. JAIN

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES AND RESEARCH

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE

OF MASTER OF SCIENCE

DEPARTMENT OF COMPUTING SCIENCE

EDMONTON, ALBERTA

FALL, 1977

THE UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research, for acceptance, a thesis entitled "Computer Matching of Stereo Pictures" submitted by Rakesh K. Jain in partial fulfilment of the requirements for the degree of Master of Science.

ACKNOWLEDGEMENTS

I wish to convey my deepest thanks to my supervisor, Dr. L. K. Schubert, for his excellent guidance throughout this research. His sincerity and encouragement was a constant source of inspiration for me. The suggestions and critical comments by the committee members, Dr. Davis, Dr. Dawson, Dr. Sampson and Dr. Wilson were of great help, and I thank all of them. Thanks are also extended to M/s Surender K. Kenue and Gerry Tychon for their assistance in dealing with the Picture Operating Systems (POPS). The financial support by the National Research Council of Canada under Operating Grant A-8818 and the Department of Computing Science by teaching and research assistantships during the course of this research is also gratefully acknowledged.

ABSTRACT

In this dissertation, we present some techniques for finding corresponding points in a pair of stereo pictures. We place major emphasis on the efficiency and accuracy of these techniques. We start with criteria for selecting areas which are informative and therefore suitable for matching through exhaustive search. For any such target area, we first find potential matches through a newly developed closeness measure, the "up and down correlation". We propose some search pruning techniques as well. We grow regions of constant displacement using a modified version of a standard region growing algorithm. We present a new criterion for attributing displacements to pixels within a matched area based on the presence of a grey level jump at the pixel and on the sign of the pixel's contribution to the correlation. We propose some special measures for resolving the ambiguity in those regions where pixels are matched at more than one displacement. We also discuss certain problems which are not fully solved and deserve further investigation.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1

Introduction

In the past, interest in stereo picture processing arose mainly in the context of aerial survey and reconnaissance applications. Topographic information was obtained with the aid of manually guided stereoplotters. More recently, attempts to build mobile robots or digitally ccntrolled "hand-eye" systems have led to extensive research in "computer vision", including the problem of computer depth perception using stereoscopic images. Potential applications of this work, besides automation of traditional methods of analyzing aerial photographs, include vision systems for motorized robots and planetary explorers [Levine, 1973]. Automation and speed of depth perception are crucial in these latter applications.

Several researchers have worked on the stereo picture processing problem. Hannah [1974] was mainly interested in pruning the space searched for corresponding regions. Levine's [1973] approach involved an adaptive method of adjusting the "window" (small rectangular region) size and included several ways of delimiting the search space on the basis of certain assumptions about the scenes. Thompson's [1975] work was an extension of Hannah's wcrk. In addition

he proposed some correction factors which can be applied in matching objects distorted because of differences in angles cf view.

Computer depth perception is achieved by solving two subproblems, the "correspondence problem" and the "triangulation problem". The correspondence problem involves lccating an appropriate area in one picture and finding the corresponding area in the second picture. The triangulation problem involves the basic triognometric and numerical techniques for locating the corresponding object in a 3-D scene on the basis of two matching regions in the stereo pair. In this thesis, we deal with the first subproblem, the "correspondence problem". We describe some techniques for determining corresponding points accurately and quickly in a stereo pair.

In all previous approaches, a window is chosen as a target (area for which match is sought) only if it contains some nonuniformities, so that it is distinguishable from its neighbourhood. The measure of nonuniformities is grey level variance. However, it is not clear how to choose a threshold on variance for any given picture, since the variance depends on the choice of grey level scale. The idea of attaching the displacement only to the centre of a matched window is also questionable, for reasons to be explained in chapter 4. Also in most of the previous approaches finding the corresponding regions involves the calculation of normalized cross correlation at a large number of points in

the second picture. Hannah [1974] proposed the use of "similarity" to restrict the calculation of this time ccnsuming operation to a small number of points. Even this approach is not very efficient, since it requires repeated calculation of a number of statistical features for different areas in the second picture. To avoid these repeated calculations would require several times the storage for the picture itself.

In chapter 2, we examine past research work in stereo vision, pointing out achievements and shortcomings of different approaches. In chapter 3, we deal with the selection criteria for obtaining those target areas which are worthy of matching through exhaustive search. In chapter 4, we discuss methods of finding a unique and accurate match for a window in the first picture. First of all we detect potential matches through a newly developed fast correlation method based solely on the sign of the up and down variation of grey level across a region; we confirm the best match (if any) through traditional correlation techniques. We propose some ideas for pruning the search for windows with high "up and down correlation". We restrict the search on the basis of assumed maximum total and relative displacements in the two directions. We have also implemented a method for skipping a certain number of rows in searching for potential matches. After confirming the match through normalized cross correlation, we attribute the displacement essentially to cnly those pixels at which there is an appreciable grey

level gradient and which contribute positively to the normalized cross correlation.

We also propose a fast way of extending regions of constant displacement. We discuss certain methods which help us in picking the most reliable displacement for those pixels which are matched at more than one displacement. These heuristic methods involve comparing the two correlations at which a pixel is matched.

Our main contention is that the matching algorithms of chapter 4 are quite efficient. The number of operations in rejecting or accepting an area as a potential match for an M1 by N1 window is O(N1) in the worst case. The modifications to the existing region growing algorithm introduce a saving factor of O(M1) or O(N1), depending on the direction of growth. The idea of allowing a point to be assigned different displacements and choosing the one which corresponds to the maximum correlation gives more reliable displacement values, especially near depth discontinuities. The method of selecting target areas of chapter 3 is simpler and more general than methods based on variance.

The following symbols and terms will appear frequently throughout the thesis.

Symbols	Meaning
*	Multiplication
**	Exponentiation
lcg2 x	Logarithm of x to the base 2
O(f(M))	Bounded by a linear function of f(M)

Terms	Meaning
Candidate area	A tentative match in the right picture for the target area.
Depth edge	An edge across which there is a depth discontinuity. (not to be confused with a grey level edge, see "edge point")
Displacement	If (IXL,JYL) are the coordinates of a pixel in the left picture and (IXR,JYR) that of the corresponding pixel in the right picture, then horizontal and vertical displacements of the pixel are respectively given by (JYR-JYL) and (IXR-IXL).
Left (or first) picture	Picture for whose windows we seek matches in the "right" (or second)

picture.

Edge point	A pixel at whose right or lower boundary there is an absolute grey level change of 2 or more.
Picture	An M by N matrix representing a picture. Each element depicts the grey level (0 to 63) intensity of a small region.
Pixel or Point	An element of a picture.
Target area	A window in the left picture whose match is being sought.
Window	A small M1 by N1 region.

Convention

In any picture A(I,J), subscript I will increase downwards and J to the right.

Chapter 2

History of Past Research

The computer stereo vision problem has become one of major importance in the field of computer picture processing. Automation was desired in many depth perception tasks (e.g., aerial survey of remote scenes, planetary exploration etc.). At first, (around 1968) researchers were just concerned with effectiveness of depth perception algorithms rather than with their efficiency. Later, more and more efficient techniques were developed. Pruning of the search space, compression of picture data, and pre-selection of areas for matching improved the speed of the matching algorithms. Hart [1968] presented a simple approach for finding corresponding points and determining their 3-D locations. Hannah [1974] developed more efficient and accurate techniques for matching two stereo pictures. Levine [1973] used an adaptive approach for finding the depth contours of stereo images. Thompson [1975] presented a methcd for correcting errors due to different kinds of distortions. Nevatia [1975] used several intermediate views instead of the usual two for processing in order to improve the accuracy as well as the speed of the matching. Barnea and Silverman [1972] proposed a sequential decision

algorithm which rejects bad matches before considering all pixels within the regions being compared. The contributions of these authors to the solution of the "stereo vision problem" are now outlined in greater detail.

2.1 P. E. Hart [1968]

Hart was among the early researchers to tackle the problem of "stereographic perception of 3 dimensional scenes" by a computer. The work was carried out in 1968-69 at Stanford Research Institute as part of the SRI automaton project.

Hart discussed two problems, the "triangulation problem" and the "correspondence problem", separately. Hart dealt with four types of errors which can be present in locating a point in 3-dimensional space after finding the corresponding matching point pair. The four are (1) camera pan angle error, (2) camera tilt angle error, (3) quantization error, (4) correspondence error. The first two are due to uncertainty in the knowledge of relative camera orientations. The third error is due to the fact that one deals with digitized pictures not with "analog pictures". The fourth error is due to finding incorrect corresponding points.

In order to calculate the error due to digitization, two pictures (of a type not known to the author) were generated and an object point was replaced by the nearest

pcint on a 120 by 120 grid. Applying the triangulation to these points allowed the calculation of the quantization error. Similarly errors due to relative pan and tilt angle differences were calculated by introducing pan and tilt angle errors and applying the triangulation procedure.

For calculating the correspondence error, again, two pictures were generated, a particular object point was chosen and its true corresponding point was chosen by visual inspection. The triangulation procedure, as usual, was applied after displacing the image point in one of the pictures by some specified amount. The difference between the computed point and true point in 3-D space yielded the correspondence error.

Hart performed three series of experiments. The first two series were concerned with isolating and measuring the effects of triangulation and correspondence error. The third series determined the overall effectiveness of the laboratory stereo system.

The results of the experiments included graphs of fractional error against the ratio of range (distance from the base line) to base line (line between the lens points for two positions of the cameras). The errors due to quantization, tilt and pan angles were kept fixed. Hart found that errors due to pan angle and due to quantization were more severe when the range to base line ratio was large. Also pan angle was the greatest contributor to the triangulation error. The error due to tilt angle was

independent of the sign of the difference.

Unlike later researchers, Hart did not concentrate on developing efficient search techniques for finding corresponding points and selecting appropriate areas which are worth pursuing for matching. He used the conventional way of matching through exhaustive search applying normalized cross correlation at every point. Apparently, his sole concern was with the behaviour of different errors.

2.2 M. D. Levine et al. [1973]

At the Jet Propulsion Laboratary of the California Institute Of Technology a team of researchers undertook a project partially directed towards a solution of the problem of "visual perception" by a robot. A pair of stereo pictures were taken with a system consisting of 2 cameras having horizontally aligned and parallel optical axes. Thus, there was no relative vertical shift in the two pictures. The goal was to obtain an accurate and complete depth map of the scene pictured. Since the robot research was directed mainly towards the development of a planetary explorer, some Californian remote field scenes were used as surrogates.

With the above type of camera system, the set of points in the second picture which are the candidates for a match fall along a horizontal line. Thus, the number of positions to be searched is just a function of one of the picture dimensions. For matching, an adaptive window size was used.

This was thought to be desirable because of the fact that if a small window is used, the probability of accidental false matches is increased; on the other hand large windows result in poorly defined edges.

A heuristic approach for selecting reference points (tie points) was adopted depending on the image data. The idea was to choose fewer points in smooth areas and many in areas where the texture was rough. In effect, uniform regions were ignored until neighbouring information-rich regions were processed.

The algorithm begins at the bottom of picture A (target picture) and processes every pth row (p is some constant chosen by the programmer) to the top. The processing includes selection of tie points for the row being processed and finding their displacements. To find a tie point, the local variance of the window centred at that point is calculated. The point becomes a tie point if its variance is maximal relative to all unprocessed points of the current row. If this variance is greater than some threshold, a small window is chosen; otherwise a large window is chosen.

Suppose that the Lth row is currently being processed. Some points in this row are excluded because of possible occlusion due to the following assumption:

Occlusion assumption: r successive pixels are assumed to be occluded at a point where there is a jump of size r in retinal disparity.

For any tie point a search limit in the second picture

is specified according to local context and knowledge of the world model. As the processing proceeds from the bottom row to the top row, the left hand and the right hand search limits for points of the current row are assumed to depend on disparities found in the previous rows. These restrictions narrow down the search limits. The calculations of search limits involve the following assumptions:

Proximity assumption: For any row l, points in row (l+p) will have approximately the same retinal disparity, if these points are part of the same object.

In order to ensure that points in row l and row (l+p) belong to the same object, a texture comparison of the local region centred at (l,j) and at (l+p,j) is made. If there is evidence of an edge, the search limit for the row (l+p) is reset some prefixed default value for that row.

Ordering assumption: For any row l, the order of points in the first picture is same as in the second picture.

Default assumption: The point most remote from the viewer in an upper row is at least as far away as the points in lower rows.

Levine applied the alogrithms to an m by m subset of points of a larger M by M picture and magnified the depth information by means of interpolation. He didn't explain the kind of interpolation used.

Levine's major contribution towards the solution of

stereo visicn problem is the narrowing down of the search space in matching. However, his pruning techniques involve assumptions which may not always hold. For example, the Ordering assumption fails if two objects differing greatly in depth lie close to the line of sight and overlap the same row (for example, a nearby telephone pole may lie to the right and left of a window of a more remote building, as seen in the left and right images respectively). The Default assumption will fail when some over-hanging objects (e.g., branches of a tree) lie near the camera. They will occupy the upper portion of the picture, but will have minimum distance. The idea of making search bounds narrower after calculating the displacements of some points is sensible, but there is no criterion for assessing the reliability of a match. The highest correlation does not always lead to the correct match, and if incorrect, one match can have adverse effects on the neighbouring regions' search bounds. His criteria for making the window size adaptive to local variance will work well in high and low frequency regions. But Levine did not propose any method of detecting one or a set of parallel straight edges through the target window, which are prone to ambiguous matches.

2.3 M. J. Hannah [1974]

Hannah was concerned with finding a unique area in one picture corresponding to an area in the other picture. She proposed and implemented techniques to improve the efficiency of matching, and found a method for extending regions of constant displacements. She dicussed some selection rules for choosing a suitable target area for matching. In view of the presence of offset and scale factor (the respective constants by which all pixel grey levels are shifted and multiplied when seen under different light), she used normalized cross correlation between a target area and candidate area.

The idea of making the search space as small as possible without missing the correct matches was a major issue in her research. The following are the search pruning techniques she employed:

2.3.1 Gridding

If a surface is formed by plotting the correlation as a function of position of candidate areas in the neighbourhood of the matching candidate area, one sees that the correlation gradually falls off from the maximum value. The values in the neighbourhcod of this maximum normally lie above the significance threshold. This is because of the local similarity (both grey level and depth) of most of the target areas.

If the width R of this neighbourhood (over which the correlation lies above the threshold) is chosen as a skip factor (i.e., the horizontal and vertical distance between attempted positions for match) one can still expect a match to be obtained at a position overlapping the position which matches best. Now searching finely in the vicinity of this pcint, one can find the exact match. Employing this technique, one can save a factor of R^2 in searching.

2.3.2 Reduction

Reduction means compressing the picture matrix. To the extent that important grey level information is not thereby destroyed, the technique of reduction can be used to realize savings in search. A new pair of pictures is formed out of the original pair by replacing each m by m square of pixels by a pixel whose value is the average over the square. Areas are then matched as usual. Finally, a fine search is carried out in the original picture in the vicinity of best matches found in the reduced pictures. This technique intoduces significant savings in exhaustive search. How far the pictures can be reduced depends on the frequency spectrum of the picture. If the major part of the information in the picture lies in features at most p pixels wide, the picture should not be reduced beyond a factor of p.

2.3.3 Similarity

Ideally, if two areas match then some statistical measures (mean, variance, etc.) calculated over them should also be similar. In order to implement the technique of similarity to reduce time, a vector of statistics for the target area is calculated. Promising candidate areas are those whose statistical vectors are close to that of the target area. The measure of closeness here is a weighted distance metric (e.g., in a vector of n statistics closeness could be distance in the Euclidian space of dimension n). This measure may be much easier to compute than normalized cross correlation. Out of these promising areas, the K most promising areas are chosen for further processing. In the neighbourhood of each of these, the normalized cross correlation is then used to find the best match.

If the similarity vectors corresponding to all the regions in the second picture are stored, very significant computational savings can be obtained, but this requires more memory. There is some tradeoff between space and time, which the experimenter can decide beforehand. Also, a larger number of complex statistics is likely to be suitable for restricting the number possible matches but may be expensive to calculate. Hannah suggested that it is not necessary to store the vector of statistics for every point in the second image, but only at the nodes of an m by m grid superimposed upon the second image. After weighted distances at these points have been found, the K most promising areas are

chosen and normalized cross correlation applied in their respective neighbourhoods. She obtained considerable time and storage savings by means of this approach. Hannah concluded that the similarity technique may not always work. If the pictures are quite homogenous in texture, all areas will be similar and the number of promising candidate areas will be too large. A comparison of results using Hannah's similarity measure and the author's closeness measure will be presented in chapter 5.

2.3.4 Camera Model

Hannah made use of the knowledge of the orientations and positions of the two cameras in constraining the search to a very narrow band, or even to a line in the second picture. To do this, a target area's centre point is projected through the first camera as a ray in 3-D space. The actual point in 3-D space corresponding to this image point must lie on this ray. This ray is now projected back into the second camera forming a line segment in the second image plane. The matching area's centre must lie on this very line. If some errors are suspected in the camera model, the search can still be restricted to a narrow band around this line. If the picture size is M by M, the savings compared to unconstrained exhaustive search amount to a factor of M.

Hannah suggested that it is possible to derive the

camera model with the help of a number of matching point pairs. Theoretically, if one has N constraints in the camera model, and N constraints in the form of matching point pairs, one can find a closed form solution for the camera model. But these constraint equations, because of possible errors in matching point pairs, may not yield a reliable sclution, so instead, the unknowns are approximated by least square techniques.

2.3.5 World Model

Besides knowing the camera model, if one knows the location of some object such as the ground plane, then it is possible to predict quite reliably the matching window positions for a visible portion of that object. The ray from the first camera will intersect the object at some point, which can be projected back into the second camera giving the exact centre point of the match. Sometimes a little bit of information of this type can constrain the search procedure for other objects as well. For example, knowledge cf the position of the ground plane limits the depth at which an object below the horizontal can lie [Falk, 1969].

Hannah used a reasonable assumption about the world, the "continuity assumption". Under this assumption if areas A and B are adjacent in the first image their matches are probably again adjacent in the second image. So, one needs to search the whole picture only if there is no adjacent

match available and the current area is not mergeable into the same chunk.

It is easy to observe that certain target areas are not good for matching. These are the areas which contain no or very little information. Hannah found that areas for which the peaks in the 2-D grey level autocorrelation function are neither too flat nor too sharp are best for matching. If the peak is too flat then it is clear that the area lies in a very smooth region; and if the peak is very sharp then the area is going to be hard to match. Also, any target area containing a linear edge is prone to give ambiguous matches, since it would match all along that edge. She detected regions containing parallel linear edges only by fitting a bivariate normal distribution to the autocorrelation function and then checking if the peak assumes the form of a ridge. If it does, the ridge must be in alignment with the parallel edges and the target area is not considered for matching.

The choice of the best match also involves checking for ambiguity of the match (more than one match at isolated points). If such a situation exists, the target area is considered unmatchable until it is mergeable with some region when regions of constant displacement are grown.

Hannah observed that the correlation surface in the neighbourhood of the matching area closely resembles the autocorrelation surface of the target area. She obtained points on this surface by moving the window in eight

different directions from the centre of the target. She suggested that the maximum of these 8 autocorrelation values should be taken as the threshold for accepting the match. She calculated and used an empirical threshold which is equal to the correlation between the target window and a distorted version of itself [Hannah, 1974]. She found that this threshold accepts 98% of of correct matches and rejects 99% of erroneous matches.

We have already mentioned that the continuity assumption helps in limiting the search space for the target areas when adjacent matches are available. After having found a reliable match for the target area centred at (IX,JY), one can expect four adjacent areas centred at (IX+1, JY), (IX, JY+1), (IX-1, JY), (IX, JY-1) to match with corresponding adjacent candidate areas. Once one of these expected matches succeeds then areas adjacent to it are similarly tried, etc.

Sometimes it is desirable to match areas which are not rectangular in shape. To achieve this it is important to be able to vary the indices in some easy way irrespective of the shape of area. Hannah used the idea of masked correlation to tackle this problem. The mask was a rectangular template which was filled with 1's in the area of interest and with 0's elsewhere. Areas were then compared as rectangular windows but considering only those pixels with nonzero template values.

Hannah's overall approach was aimed at making the

search for matches efficient. The techniques she proposed and implemented were very useful and influenced later researchers' work. According to her, similarity, gridding and reduction respectively give savings factors of 9, 50 and 100 over the normal way of calculating normalized cross correlation at every point. The savings due to use of a camera model are a factor of M (width of the picture). The region growing algorithm saves a significant amount of time when the pictures contain regions of approximately constant displacement (depth). Like other researchers, Hannah employed the questionable criterion of assigning the displacement of a window to the point at the centre of a window. She applied her matching algorithms to outdoor scenes and displayed some of the results by drawing frames around corresponding regions in the stereo pair. It is not possible to tell from these results how reliably matches were obtained, or how frequently and under what conditions errors occurred. Moreover, Hannah's complete matching algorithms were not fully implemented at the time of publication of her thesis.

2.4 C. Thompson [1975]

Thompson also attacked the problem of depth perception with stereo vision and was even more interested in 3-D modelling of the world. In his research at Stanford University, he extended the work of Hannah. While

experimenting with different matching techniques designed by Hannah he came up with a new empirical threshold for accepting matches. He noticed that the highest correlation for the matchable target areas is normally the average of 1.0 and the autocorrelation threshold. According to Thompson this is possibly due to an average error of ±1/2 pixel in finding the exact match. He also saw that the probability of finding a correlation maximum less than the autocorrelation is quite small. On the basis of this, he used another threshold for accepting the correlation, given by k+(1-k)*autocorrelation ($0 \leq k < 1$). Thompson suggested that k can be varied to make the criterion less or more strict. He saw in his experiments that k=0 discards only very unlikely matches, k=.5 discards half of the good ones. The former is appropriate for a global search. The idea is to avoid false matches while keeping the probability of not missing the correct match at a high level. The latter is used in searching locally for a match corresponding to the target area immediately adjacent to a previously obtained good match. A few mismatches were obtained with the above threshold; all had very low autocorrelation (less than .5). To overcome this type of difficulty, Thompson raised the threshold for low autocorrelations by changing it to Max{(k1+(1-k1)*autocorrelation), k2)}. k1 and k2 are constants.

Because of the the change in angle of view, there is every possibility of nearby objects getting distorted from

one picture to the other. Thompson proposed methods of calculating and applying corrections to distorted areas for matching. He applied two types of corrections, "uniform scaling" and the "directional scaling".

Uniform scaling is meant to neutralize the distortion because of a difference in the distances of an object in the scene from the two cameras. For example, if an object is twice as close to camera A as to camera B, it will appear to be approximately twice as large as in camera A. To correct this disparity, the target window (in the picture A) should be made twice as small in each direction as the candidate window (in picture B). Thompson implemented uniform scaling as follows. If at any stage the candidate window at displacement (DI,DJ) is assumed to be the real match, then on the basis of this assumption one can calculate the distance of the object from the two cameras (provided the camera model is known). Now it is simple to calculate the ratio of the window sizes.

Directional scaling is used to correct the distortion due to different orientations of the face of an object relative to the viewing angle of the cameras. To see how distortions appear because of different orientations of the two cameras, one needs to look at two angles. Consider the plane formed by the lines from each camera centre to a point on an object. A small portion of the face of an object may be represented by a square surface whose centre is at the intersection, and whose orientation is described by two

angles. The first is the "dihedral angle" between the plane and the square. The ground is normally at a small dihedral angle. Objects like doors, trees are at a large dihedral angle (about 90 degrees). The second angle, the "normal angle" is the angle between the projection of the surface normal of the square onto the plane (defined above) and the line from either camera to the square. This angle is different when measured with respect to the two camera positions.

Directional scaling takes care of the relative size changes due to different normal angles. Thcmpson proposed that a reasonable correction for this distortion is to vary the aspect ratio of the target window. Distortion due to normal angle is going to be large for nearby objects. Obvicusly, the correction is not possible unless one knows the angle of inclination of the face of the object under consideration. Thompson believes that it is possible to reduce the number of different possible aspect ratios by imposing an upper limit (say 89 degrees) on the normal angle. For coping with the distortion due to dihedral angle one could, for example, match a square window to a parallelogram. Thompson pointed out that for low objects lying on the ground plane, this correction would be very useful.

Realistically, these corrections (for dihedral and normal angles) are not easy to apply. In addition to the requirement of limiting the range of normal and dihedral

angles, one has to try corrections for different angles, which may be a very time consuming approach.

2.4.1 3-D Modelling

Thompson's work also included an algorithm for 3-D modelling. After applying point-pair matching routines, a program (GEOMED), which is capable of displaying 3-D line drawings from any orientation, is called. Lines are drawn between all 3-D points that correspond to neighbouring target areas in the picture A. Since this produces too many lines to make any sense of, unimportant lines are deleted by the following rules:

1. When a pair of lines cross, delete one of them.
2. When four non-colinear points are approximately coplanar, delete "diagonal" or "interior" lines.
3. When a line connects two "objects" (see below), delete it.

The objects are separated by sorting all 3-D points by their coordinates in the picture, and then drawing lines between all points that are adjacent in the picture (horizontally, vertically and diagonally). The upper left hand matched point of the picture is assigned to object #1, then an attempt is made to extend it by including more and more points (moving only along lines connecting adjacent points). As points accumulate in one object, a surface is eventually determined. Components of the change in depth

with respect to the two picture coordinates can be approximated. The merging requirements for points is that their Z-coordinates (depth) match the extrapolated Z coordinates of the surface plane within a wide error bound (about 50 times the estimated error in Z coordinate). When no more points can be included in object #1, object #2 is started with the upper leftmost point that hasn't been included in any object yet.

Thompson developed an algorithm for face separation as well. First, all posible non-overlapping triangular faces are constructed, whose three vertices lie on the same object. The adjacent triangles are assigned "faces" starting with the upper left most triangle. If IA and JA are the picture coordinates in picture A, then the rates of change of depth with respect to these coordinates are calculated by the first triangle assigned to the "face"; successive triangles must share an edge, hence, two vertices with the "face". The depth coordinates of the third vertex must lie within a narrow bound (about twice the estimated error in Z coordinate) of extrapolated depth coordinates. Thompson admitted that discrimination of objects on the basis of descriptions obtained in this way was not handled very successfully by his algorithms.

Thompson's achievement lies in developing a new empirical threshold for matching by cross-correlation and proposing corrections for different types of distortions. His 3-D modelling algorithm represented the first attempt to

obtain meaningful scene descriptions from the matched stereo images.

2.5 R. Nevatia [1975]

Up to now, we have been discussing the common approach of using two images of a scene separated by a chosen angular difference. It has been found that error in depth measurement depends directly on the angular differences. A larger angle will give more accurate information; but displacement increases proportionally resulting in the need for more extensive search and in increased distortion; hence chances of error are increased. Nevatia conducted some experiments at the Stanford AI laboratory to show that, if intermediate views between two stereo images are used, then searching becomes less time consuming and more accurate.

He obtained the progressive views of a scene by making the environment move with respect to the camera. This type of set-up is equivalent to a mobile robot, which updates its view of its enivironment during its motion. In his experiments the objects were lying on a turntable which could be rotated by specified increments while keeping the camera stationary above the turntable. He rotated the turntable in steps of half of one degree.

When several views are available, then on the basis of the assumption that the width of the required search band is directly proportional to the angle between the two views,

Nevitia suggested that search can be reduced considerably. Suppose there are k views in all, and displacement between the two extremes must lie within a band n pixels long and m pixels wide. Then the search for correspondence between two adjacent views can be limited to a band of (n/k-1) pixels long and (m/k-1) pixels wide. Let the successive views be called view 1,..., view k. Displacements between extreme views can be determined by chaining these successive views, and the search at each step can be limited in the above fashion. Search in this manner realizes a saving factor of (k-1), as compared to the method of direct searching between two extreme views.

In some cases, the width of the band to be searched is not directly propotional to the angle between two views. If this is the case, no time savings result. Nevatia claimed that even in this case, the use of progressive views is advantageous, because of its greater reliability. Two different regions in one view are sometimes similar to the same region in another view, so that choosing one region based only on small differences in the similarity measure will occasionally lead to a incorrect choice. Using progressive views, search at each step is limited to a small region, and thus the probability of finding an erroneous region in another part of the image is greatly reduced.

2.6 Barnea and Siverman [1972]

Ordinarily, when the correlation between a target window (a window whose match is being sought) and a window at some reference point is computed, all pixels within both windows are considered, even if the reference point is far from the correct point of registration. Barnea and Silverman suggested that accuracy is required only for promising reference points.

Let S and W be the window at the reference point and the target window. There are M^2 points (M is the size of the square window W) of S which are to be compared with M^2 corresponding points of W. Define each pair of corresponding points as a windowing pair.

SSDA (the Sequential Similarity Detection Algorithm) reduces the computational work by performing a sequential search, which may be terminated before all M^2 windowing pairs for a particular reference are tested.

In one version of SSDA which is called the "Constant Threshold Algorithm", windowing pairs are selected for comparision in a nonrepeating order. Whatever the measure of closeness is, normalized or unnormalized, the cumulative error between windowing pairs is calculated. This error is monitored at fixed intervals by testing it against a preset constant threshold T. When the accumulated error exceeds T at test N, the processing is aborted and N is recorded. Reference points for which N is large are considered as points of promise.

If a suitable value of T is selected, many fewer than M^2 tests are required for those reference points which rapidly accumulate error. The reference points for which error accumulates quite slowly are likely to be the candidates for registration and are processed further.

Another version of SSDA is the "Monotonic Increasing Threshold Sequence Algorithm". It is clear that the growth curve for the error at a particular reference point is a monotonically increasing function. It is the average slope of the growth curve that is important in the determination of a threshold crossing criterion. It therefore seems reasonable to replace the constant threshold T by a monotonic increasing function Tj. The sequence Tj should have sufficiently high initial values so that enough windowing pairs will be used to determine the trend of the growth curve reliably. In their report Barnea and Silverman specify several such increasing threshold functions.

With these measures for selecting reference points for further processing, they found that a time saving factor of up to 50 can be realized. But this saving also included the saving due to the fact that they used the sum of absolute differences as the closeness measure instead of the normalized cross correlation. For this closeness measure, any global or local change in contrast (gain) or offset between the two pictures can lead to rejection of the correct match.

In summary, the following three shortcomings are common

to all of the approaches discussed above:

1. Region overlap: When a unique match for a window is found and extended, part of the resulting region may also be matchable at a different displacement during the growing of some other region. Hence the problem arises whether to retain the old displacement or to replace it by the new one. Apparently no previous researcher has dealt with this problem.
2. Displacement assignment: When the match for a window is found, which portion of it should actually be considered matched and which disregarded? The common answer of considering the centre of the window matched has already been criticized.
3. Selection of informative windows: We have already explained the weakness in the use of grey level variance as the measure of information in a window.

Chapter 3

Selection Of Target Areas

Prior to searching for the match for any window, it is desirable to see how much and what kind of information it ccntains. It is fruitless to seek a match for a window lying in a more or less uniform region (few sharp discontinuities in grey level), for it will match well with all windows in the uniform regions of the second picture. For a window to ccntain some information there should be at least some minimal number of grey level discontinuities.

As was seen in the previous chapter, a common measure of information in a window is its grey level variance. If the variance crosses some threshold, the window is considered matchable; otherwise it is rejected.

There is an implicit weakness in this selection criterion. Consider two windows $X(i,j)$ and $Y(i,j)$ ($1 \le i \le M1$, $1 \le j \le N1$) containing exactly the same pattern with different scale factor and offset, i.e., $X(i,j) = a*Y(i,j) + b$ for all i and j. (a is called the gain and b the offset). Clearly, the variance over the window X is a^2 times the variance over the window Y. Thus the measure of amount of information in a window based on variance is not normalized with respect to gain.

We have therefore chosen to base our measure of information in a window on the number of grey-level jumps which exceed some very low threshold. The threshold is chosen so that jumps exceeding the threshold are unlikely to be due to discretization error or extraneously introduced noise, i.e., they should correspond to actual intensity changes in the original scene. In several different pictures taken under different lighting conditions the digitized representation of "smooth" regions (such as plain walls and table-tops) were examined. Local fluctuations of digitized grey level in such regions can be attributed to hardware noise and digitigation error. In all cases such fluctuations were found to be of magnitude≤2. Therefore a threshold of 2 appears to be appropriate for the hardware used in the present study.

However, a count of grey level jumps is misleading as an indicator of "matchability" in case a window contains only parallel straight edges. Such windows will lead to ambiguous matches, since any translation of the window in the direction of the edges leaves the pattern within the window invariant. We will assume that multiple parallel edges within a window occur with negligible frequency, but will discuss a method of eliminating windows with single straight edges as targets for matching by exhaustive search. Before proceeding further, we define the following terms:

A pixel is said to precede a <u>horizontal jump</u> if the absolute difference between its grey level and that of the

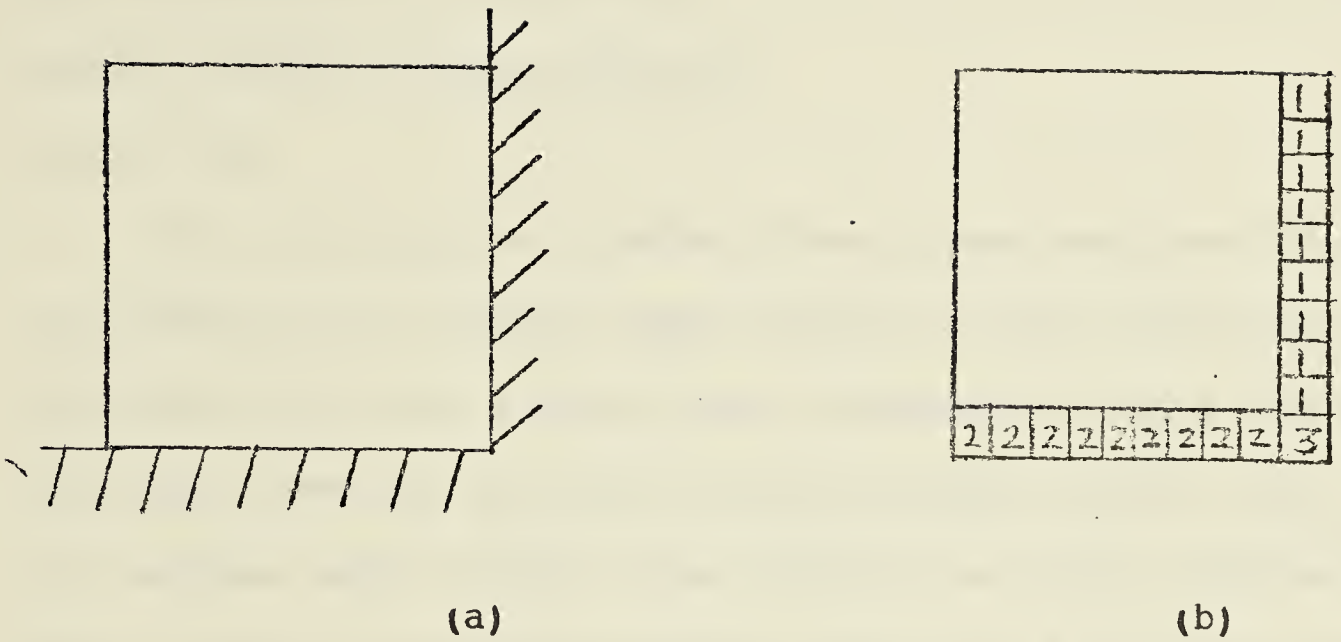
neighbouring pixel to its immediate right exceeds a fixed threshold. Similarly a pixel is said to precede a vertical jump if the absolute difference between its grey level and that of the neighbouring pixel immediately below it exceeds the threshold.

The edge code of a pixel is 1 if the pixel precedes a horizontal jump but not a vertical jump; its edge code is 2 if it precedes a vertical jump but not a horizontal jump; its edge code is 3 if it precedes both a vertical jump and a horizontal jump; otherwise its edge code is 0.

A pixel is said to be an edge point if its edge code is nonzero.

When counting the number of edge points of the picture, which contribute to grey level variation in a window, we should consider only those edge points which are determined by a pair of pixels within the window. Consider those pixels in the bottom row of the window whose code values are 2 and those pixels in the rightmost column whose code values are 1. The actual grey level variation due to these edge points lies outside the window (see Fig. 1).

Based on the above discussion, we can say that a window is matchable if it has enough edge points and these do not all lie along a straight edge. These conditions are relaxed (for reasons to be explained in chapter 5), when growing the region of constant displacement. When no adjacent match is available or a window is not mergeable with any region of



(a) (b)

Fig. 1. A window (a) lies in a completely uniform region but has edge points in last row and last column (b).

known displacement, both conditions are necessary.

To detect the presence of a single straight edge through a window, we fit a straight line through the edge points in the window. We then check whether there are at least P edge points lying sufficiently far (say D pixels) from this straight line. If so, the window is matchable; otherwise not.

A reliable and an efficient way to determine the best-fitting straight line is the method of least squares. If the edge point coordinates in the window are ((X(i), Y(i)),i=1,L) and the best straight line is reperesented by Y=a+b*X, then the two parameters a and b (slope) are given by the follcwing equations:

```
S1=X(1)*Y(1)+...+X(L)*Y(L)
S2=X(1)+...+X(L)
S3=Y(1)+...+Y(L)
S4=X(1)*X(1)+...+X(L)*X(L)
b=(S1-(S2*S3)/L)/S4-(S2*S2)/L
a=S3-b*S2
```

If the slope is greater than 1, we can take the measure of deviation of window edge points as the horizontal distance from the fitted line; otherwise we use the vertical distance. We can scan the array of edge points (X(i), Y(i)) to see how many window edge points are lying off the line. When P points are accumulated which lie D or more pixels off the line, the algorithm terminates and the window is passed as matchable; otherwise this process continues till we decide that all the edge points lie on a straight line.

In the latter case, it is still possible that the window is matchable, viz., if the line is broken. Gaps are easy to detect, since we know the slope of the fitted line. If this slope is less than 1, then there should be at least G (smallest length of a genuine gap) successive columns in the window which are free of edge points near the fitted line. If the slope is greater than or equal to 1, the same condition should hold for the rows of the window.

These are the criteria for selection of target areas based on the amount and type of information they contain. Sometimes, it is not worthwhile to search for a window's match even though it contains enough information. For

example, areas whose autocorrelation threshold (see chapter 2) is quite low (less than .5) are likely to be matched at many isolated points in the second picture which we eventually reject as ambiguous matches. So, we need not waste time on such areas.

Since we are using up and down correlation as the measure of closeness to get promising areas in the second picture, we also reject those target areas which have a very high (liberal) up and down correlation threshold. The reason is that too many promising areas will be found.

After having found a sufficient number of edge points in a window, we need to check how many of them are still unmatched (i.e., for which we have not yet found corresponding points in the second picture). We do this just before calling the line finder subroutine. If we find that at least half of the edge points are unmatched, we accept this area as a target area for exhaustive search; otherwise not. This criterion is useful for the following reason. Even under the best of conditions, we cannot expect to find displacements for all edge points of a picture (note that a single noise point, i.e., a one pixel "speck", will introduce 3 unmatched edge points). For a window with few unmatched edge points, therefore, an exhaustive search is likely to prove futile and should be avoided. On the other hand, a window in which the majority of edge points are unmatched deserves to be considered as a target area. We can count the number of unmatched edge points in a window quite

efficiently. We just sum the number of such points in each of its columns. We retain these sums and update them in a sliding manner as we advance vertically in the processing.

Chapter 4

Matching of Pictures

The matching of two pictures consists of a series of steps until all M1 by N1 windows of the left picture have been matched against corresponding windows of the right picture, or at least until all such windows have been considered for matching. Each step involves finding a match fcr a window and then extending this to the largest possible regicn with the same displacement. In the second picture of a stereo pair, images of objects are geometrically distorted relative to their images in the first picture, the distcrtion being greatest for nearby objects. Since it is not easy, in practice, tc correct this distortion (see Thompson's work Sec. 2.4), the ratio of the distance between the two camera positions to the distance of the nearest cbjects was kept small in our work ($\leq 1/8$). With this ccnstraint we found that comparing windows of identical size and shape in the two views gave satisfactory results in the matching algorithms.

How is the match for a window chosen in the other picture? If we were able tc normalize the two pictures globally so as to to equalize the average grey level and contrast for all corresponding windows, then pairwise

ccmparison of windows would be simple. We would sum the absolute differences of the corresponding pixel grey levels and see if this sum stays within some prespecified threshold. If so, we would conclude that the two windows match; otherwise not. However, it is not possible in general to pre-normalize the pictures satisfactorily by any global transformation of grey levels. First, each picture may ccntain regions not visible in the other. Their contribution to global averages introduces an error of unknown size into the normalization. Second, the brightness of a surface visible in both pictures can be very different in the two pictures depending on lighting conditions, the reflectivity and orientation of the surface, and the change in angle of view. No global normalization scheme can take into account these local changes in grey level scale.

This problem is solved (albeit expensively) by the use of normalized cross correlation [Hart, 1968, or Hannah, 1974] to determine the degree to which two windows match. Values of the normalized cross correlation are unaffected by linear transformations of grey level scale. An autocorrelation threshold is calculated (see chapter 2) and the correlation is tested against the threshold. If the cross correlation exceeds this threshold, the match is accepted; otherwise it is rejected. We have slightly modified Hannah's way of calculating a threshold. She distcrts a window by breaking it into four triangles along the window's diagonals and moving them outwards by 1 pixel.

For the sake of simplifying the indexing, we break the window in to four quadrants and move them 1 pixel diagonally outwards (Fig. 2).

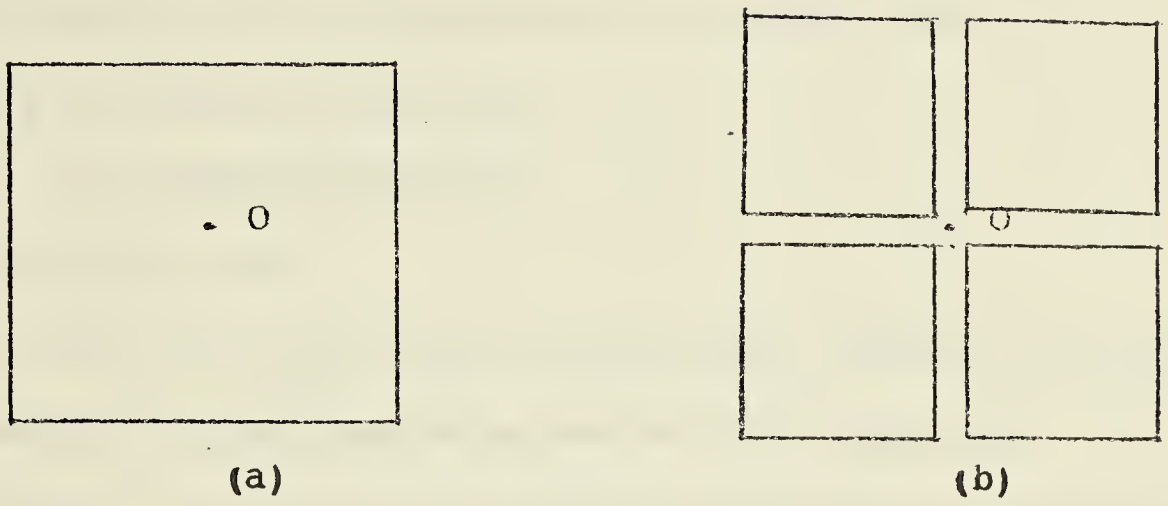


Fig. 2. Autocorrelation threshold is calculated between the window (a) and its distorted version (b).

In this kind of processing all pixels within both windows are taken into account. Since most of the time is spent in finding mismatches, the consideration of all pixels within windows is not justified. In chapter 2, we saw some techniques to avoid spending time on all points when two windows are far from matchable. We introduce the following new techniques.

4.1 Up and Down Correlation

Instead of applying normalized cross correlation at every point, we can compare gross features of two windows. Promising windows are those for which such features are

closely matched. Detailed correlation can be applied in the neighbourhood of all promising regions to confirm the match.

The gross feature vector we have chosen is based on the sums of the grey levels of columns of the M1 by N1 window to be matched; let these sums be SUM(1),...,SUM(N1). We form a ternary vector UDV of dimension N1-1 such that

UDV(I)=2, if SUM(I)>SUM(I+1);

=1, if SUM(I)=SUM(I+1);

=0 otherwise.

We call the vector UDV the "up and down vector", and call the process of matching this vector "up and down correlation". The choice of UDV for preliminary matching is based on the following intuitive considerations. First, summation over columns will decrease the effect of noise and discretization errors on the up and down variation of grey levels in the horizontal direction. Second, up and down patterns can be computed very cheaply, by the use of "running sums" in the vertical direction. Finally, up and down vectors will tend to be insensitive to small vertical shifts. Thus, in searching for up and down matches, we can often skip several rows of the picture after searching for matches in a given row[1].

---

1. Since these arguments apply equally to summation over rows instead of columns, we could presumably have used the former instead of the latter with comparable success.

As a measure of closeness, up and down correlation is superior to Hannah's measure for the following reasons. First, her method was based on a number of statistical features (i.e., variance, mean, etc.) of areas. So, storing such vectors for all areas in the second picture to avoid recomputation would require a storage whose size would be several of times that of the picture size. Secondly, it is difficult tc formulate a scale-independent criterion of "closeness" between the target's feature vector and the feature vector of regions in the second picture, because features such as mean and variance are not independent of the gain and offset. By contrast, the new method is based solely on the up and down variation of grey level in the windows to be matched, which is independent of gain and offset. Further, it is possible to compute and store efficiently the up and down vectors for all areas in the second picture (see below). We will present an experimental ccmparison in chapter 5 showing that up and down correlation leads to faster and more reliable matching than Hannah's statistical feature method.

Ideally, the UDV's cf the two corresponding windows should be identical. But, due to noise and discretization error, some discrepancies can occur and we need to set some cutoff value, within which the discrepancy may be allowed to lie. To find this threshcld, we move the window to be matched 1 pixel upward and 1 pixel downward; and we get two new UDV's. Now, we take the sum of absolute differences of

elements of these vectors from the elements of the undisplaced vector. We choose the minimum of these two sums as the threshold in up and down correlation. For windows to be of some promise, the sum of the absolute differences of the corresponding elements of their UDV's and the target's UDV should be less than or equal to the threshold. The calculation of this threshold is based on the same idea as that of calculating the autocorrelation threshold by selecting the maximum of the correlations between the target window and its eight neighbouring windows (see chapter 2).

In the worst case, we need only O(N1) operations to determine whether a window is promising. We can calculate UDV's for all window positions in the second picture using O(M*N) operations and store these UDV's in an array whose dimensions are those of the picture array. If RUDV(i,j) is the matrix consisting of all such UDV's, then the elements RUDV(i,j) through RUDV(i,j+N1-2) will represent the UDV for a window position whose left hand upper corner lies at (i,j). The savings realized by storing all UDV's is significant, since most of the time is spent in a global search to determine up and down correlation.

4.2 Skipping

"Skipping", the one-dimensional version of "gridding" (see chapter 2), can be used to make the search more efficient. After finding the up and down threshold for a given window in the first picture, we can find the "skip factor" as follows: We move the window upwards and downwards as long as the UDV's of the new position do not differ from UDV of the original window by more than the threshold. The skip factor will be the sum of distances travelled in both directions. If K is the skip factor, we need only examine every Kth row of the second picture in searching for matching up and down vectors; we can expect to get the match of the window (if any) by applying detailed correlation in the neighbourhood of a promising area.

Normally, the up and down threshold is either zero or a small integer. In the majority of cases it turns out to be zero. When the threshold is zero, then for any window to be promising, its UDV should be identical with the UDV of the target window. Thus, instead of considering the sum of absolute differences of the corresponding UDV elements, we can use equality testing, since the latter is much faster. We scan the elements of the two vectors from first to last. The moment two corresponding elements are not the same, we can reject the match. In terms of probability, let q be the probability of an element of UDV being 1 and p/2 (p+q=1) be the probability of an element being 2 or 0 (assuming equal number of 2's and 0's in the picture). For the first two

corresponding elements not to match, the probability is $(2p-3p^2/2)$. For the pictures on which the algorithms were tested (see Sec. 5.2), the value of p for windows acceptable as target areas (i.e., containing enough edge points which do not lie along a single straight line) appears to be at least 2/3. So when we start comparing two UDV's element by element, the expected number of comparisons is at most $1/(2p-3p^2/2)=3/2$.

To see the difference in the complexity of computation between normalized cross correlation and up and down correlation, let us denote the two windows to be correlated by X(i,j) and Y(i,j) ($1\leq i\leq M1$, $1\leq j\leq M1$). If M1=N1, then C, the square of the normalized cross correlation, (which is used to avoid the square-root operation) is given by the following equations:

```
S1=M1**2*(X(1,1)*Y(1,1)+...+X(M1,M1)*Y(M1,M1))
S2=Y(1,1)+...+Y(M1,M1)
S3=M1**2*(Y(1,1)*Y(1,1)+...+Y(M1,M1)*Y(M1,M1))
S4=X(1,1)+...+X(M1,M1)
S5=M1**2*(X(1,1)*X(1,1)+...+X(M1,M1)*X(M1,M1))
C=(S1-S3*S4)**2/((S5-S4*S4)*(S3-S2*S2))
```

The cost of computing C is dominated by the calculation of S1, S2 and S3, since the first factor in the denominator does not vary as we move the window in the second picture.

Now, for calculating S1 we require $M1^2+1$ multiplications and $M1^2-1$ additions. If sliding sums are used then every "window shift" needs 2 additions and 2

subtractions for updating the window sum (if M1<<M and M1<<N) by first summing in one direction and then in the other. So for S2 we require 2 additions and 2 subtractions, and for S3 5 multiplications, 2 additions and 2 subtractions. Assuming 1 multiplication and 2 additions per array access, the array accessing cost in the computation of S1 is $2M1^2$ multiplications and $4M1^2$ additions. For calculating C, we need, in addition, 4 multiplications 3 subtractions and 2 floating-point multiplications/divisions. Thus the total number of operations required is $3M1^2+10$ mulitiplications, $5M1^2+10$ additions and subtract and 2 floating-point multiplications/divisions. Integer multiplications and floating-point multiplications are respectively 3 and 20 times slower than additions and comparisons. So in applying normalized cross correlation over 10 by 10 windows we require the equivalent of about 1480 additions and subtractions. On the other hand, for up and down correlation, we need on the average only 3/2 comparisons in the (usual) case that the up and down threshold IUDTH=0. These 1.5 comparisons require 10.5 operations including the array accessing cost of one 1-dimensional array for the target's UDV and one two 2-dimensional array for the candidate's UDV. So, in the majority of cases we can expect savings of about 140 in searching.

When IUDTH is nonzero, we can expect the sum of absolute differences of the first IUDTH corresponding

elements of the target's UDV and any UDV in the second picture to accumlate to IUDTH. Since IUDTH is small (on the average 2), we can start monitoring the difference sum after considering one plus IUDTH corresponding elements and abort further computation if the up and down threshold has already been crossed. If we assume IUDTH=2, then employing this methcd of UDV matching for a 10 by 10 window, we can expect savings of about 2.5 as compared to the calculation of the sum of 9 absolute differences of corresponding elements of UDV's. Assuming absolute difference operations 2 times slower than additions, then performing similar calculations as above we need about 33 operations to match UDV's. The saving relative to normalized cross correlation in this case is about a factor of 45. Note that the savings in both the cases are not the net ones, because cross correlations are performed at potential matching points. But the number of such correlations performed is very small as compared to the total number of reference points in the second picture. Also the savings in both cases are increasing functicns of the windcw size.

4.3 Search Pruning

Searching the whole picture or even a major portion of it to find the promising areas is not always necessary. If we have some rough idea of the minimum distance of objects from the cameras (nearest objects correspond to maximum

displacement relative to objects at infinity), then knowing the camera model parameters, we can put some limit on the maximum absolute displacement and maximum relative displacement.

These terms are defined as follows. The absolute horizontal and vertical displacements are respectively the number of pixels the matching window in the right picture is shifted rightward and downward relative to the target in the left picture. The relative displacement of any two pairs of matching windows is the difference in their absolute displacements. Let AH, AV, RH and RV be the maximum absolute horizontal displacement, the maximum absolute vertical displacement, the maximum relative horizontal displacement, and the maximum relative vertical displacement, respectively. In the beginning, when no match is available, the horizontal width of the search area corresponding to a given window would be 2*AH (AH due to a possible negative displacement and AH due to a possible positive displacement). Similarly the vertical width of the search would be 2*AV. After finding at least one reliable[2] match, we can further constrain the search for future up and down matches in the following way: We keep four values VMIN, VMAX, HMIN, and HMAX in a buffer; they represent

---

2. One of the criteria of reliability of a match is whether the match is extendable by at least P full windows. After trying various values for P, we chose 5 as a reasonable value.

respectively the latest minimum vertical, maximum vertical, minimum horizontal and the maximum horizontal displacements for the reliably matched windows. Then at any stage, the horizontal search limits are {(HMAX-RH),(HMIN+RH)} and vertical search limits are {(VMAX-RV),(VMIN+RV)}. These four limits are relative to target window position. If some of these four limits lie outside the second picture range, they are set to appropriate bounds of the picture. Overall processing time can be greatly reduced by introducing search limits in this way. For example, the processing time of a 150 by 150 pair of pictures (Figs. 5a,b) was reduced by a factor of about 4 after introduction of search bounds AH=38, AV=38, RH=40, and RV=20.

4.4 Displacement Assignment

We have already discussed the criterion of assigning the displacement to a pixel (see chapter 2). A pixel is assigned the displacement (ID,JD), if a window centred at this pixel is matched with a window displaced by (ID,JD) in the second picture. This approach is unreasonable if the centre pixel of the window is not an edge point. Suppose, for example, that a boundary of some object passes through a window, dividing the window into two parts, e.g., a portion of the object and a portion of the background. Then the displacement of the window will be determined by the distance of the object (assuming that its boundary is the

dominant feature within the window). However, the centre of the window may equally well lie on the background as on the object. Thus the displacement properly belonging to the object may well be assigned to the background. This criterion also limits the chances of a point being matched, because there is only one window, centred at the point itself, which can lead to its match.

As an alternative strategy we may consider assigning the displacement to all edge points in the window. However, this raises the problem that an edge point may be present in more than one window matched at different displacements. We shall present its solution a little later. When two windows match with a fairly high correlation, most of the edge points in the windows are likely to be correctly matched, with the possible exception of very few pairs of points. The latter case is possible when most of the edge points in the windows correspond to an object at one distance while a few ccrrespond to an object at a different distance. The edge points which lie on this second object will normally contribute negatively to the overall correlation. So, an improved approach is to assign the displacement to all edge points <u>which contribute positively to the correlation</u>.

Now consider the case when an edge point is present in more than one window matched at different displacements. This type of situation is most likely to occur near depth edges (across which there is a depth discontinuity). In crder to resolve this ambiguity, we can keep some record in

the form of an array which for every edge point supplies the maximum correlation with which a window containing this pcint was previously matched. Whenever an edge point is matched, we can look at its previous correlation and replace the old displacement by the new one, if the correlation in the latter case is higher. At the same time we also update its correlation. In effect, therefore, we use the ccrrelation of a window as a measure of "confidence" in the corresponding displacement.

We conclude that every edge point should be taken as matched if it lies in at least one matched window and contributes positively tc its correlation. The most reliable displacement is due to that window position which was matched with highest correlation.

4.5 Extension of a Match

The assumption of physical continuity of objects helps in growing regions of constant displacement (see chapter 2). After a unique match is found, we can try to extend the match in all directions (left, right, up, down). Once any new window has been matched, we extend it in turn. In doing sc we form a stack of window positions around which we can grow the region and continue to grow till the stack is empty.

If we were assigning displacements to centres of windcws only, we would need to proceed in 1-pixel steps.

Thus to extend the original match by P pixels in a particular direction, we would need to calculate P correlations and autocorrelation thresholds. However, our criterion for assigning displacements allows us to consider an approach in which we attempt to move in any direction by steps equal to the dimension (DIM) of a window in that direction (vertically M1 and horizontally N1). We check whether the new window is mergeable with the region. If it is mergeable, we attempt to extend the new window as well. For each unsuccessful attempt to add a new window to the region (i.e., if the correlation doesn't exceed the threshold), we cut back the step size to half its previous value (i.e., half window within the matched portion and half outside). If the window in this position matches, we take a forward step, equal in size to half of the remaining unmatched portion. When the match is not grown by a full window, we do not grow the region around the newly matched portion. Employing this approach, we need to calculate only {(P/DIM)+log2 DIM} correlations and thresholds instead of P. Thus a saving factor of DIM can be obtained.

In an attempt to extend a match in one of the four directions, the following three cases are possible:

1. The match is extendable by a full window.
2. The match is not extendable by a full window because the extension by a full window makes either the target window or its corresponding window or both cross one of the picture boundaries.

3. The match is only partially extendable (possibly not at all) because a depth discontinuity has been reached.

In the first two cases, the correlation between the two corresponding windows is normally well above threshold, since we are probably still within a region of constant displacement. But, in the third case, this correlation may not be so high (though of course still above threshold). The reason for this is that the partial extension obtained by halving successive step sizes may protude slightly into a region of different depth. Similarly a region grown on the other side of the depth edge may protude across the edge from the opposite direction. When the two regions overlap, the displacement of one of the regions (whichever gave higher correlation) may be assigned on both sides of the depth edge, thus giving incorrect displacements to edge points on one side of the edge. The likelihood of such false assignments can be minimized by a further refinement of the region-growing strategy. After extending a window in some direction as far as possible using the above method of halving successive step sizes, we retract the window in 1-pixel steps a small number of times, computing its correlation with the correspondingly shifted windows in the second picture. Within very few steps the window is likely to reach a position entirely within a region of uniform depth. The correlation at that position should be higher than that of any window protuding across the depth edge from the opposite direction. Thus any incorrect displacements

will now be replaced by their correct values.

While extending a region in the above fashion, we must keep some record of which window positions have already been tried. In addition, we need to know the direction from which any window was tried as an extension of the region, since a part of it may have been matched by horizontal extension of an adjacent window and part of it by vertical extension of an adjacent window (see Fig. 3). Having recorded whether a

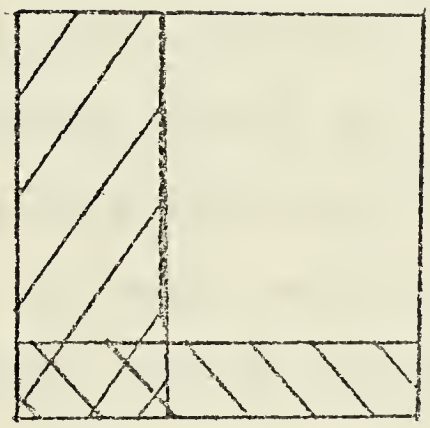


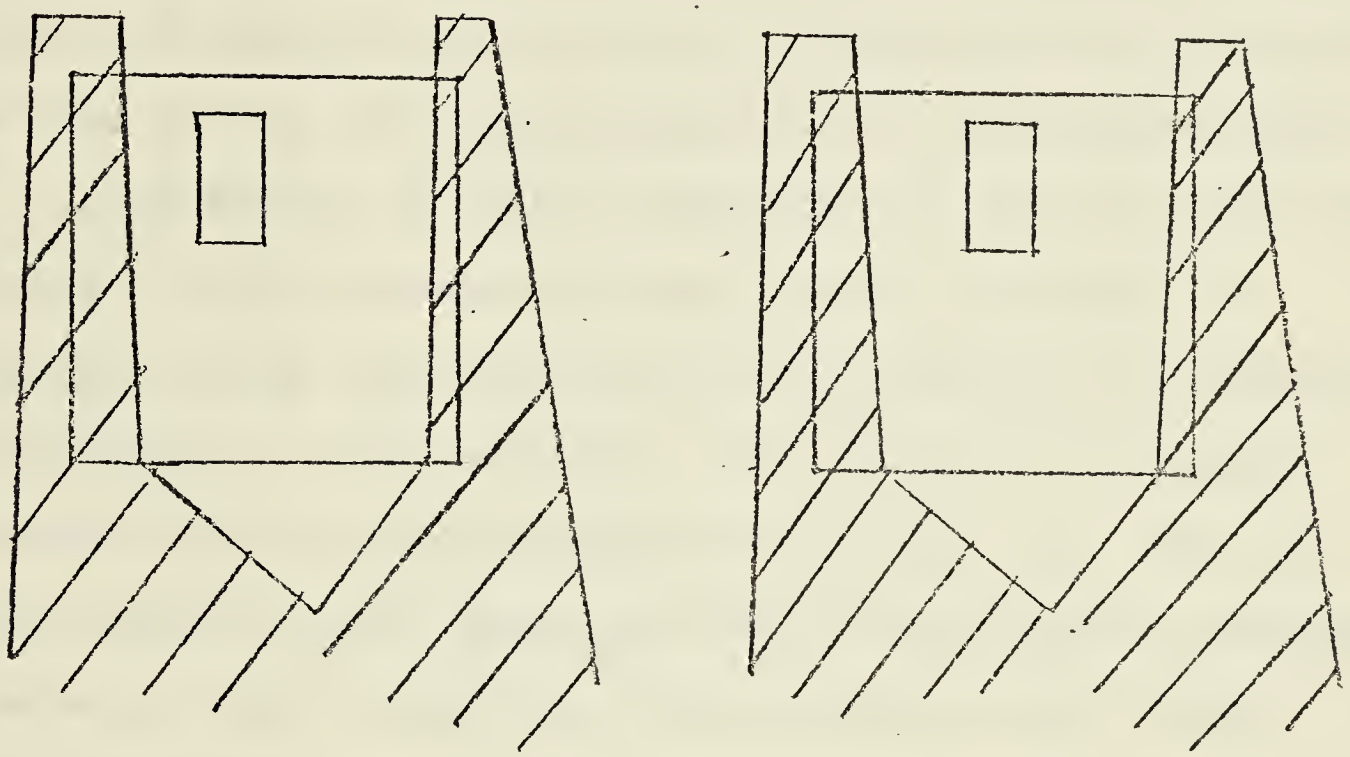
Fig. 3. Part of the shaded window is matched in horizontal extension and part of it is matched in vertical extension of a region.

window has been tried as a horizontal and/or vertical extension of the region, we use this information as follows. A window which has been tried as a horizontal extension can only be considered as a vertical extension in the future, and vice-versa. When a window is completely matched, we consider the window as having been tried as an extension in both directions.

We call the required information the extension codes. We can associate these extension codes with the upper left hand corner of the tried window positions and store them in the confidence (correlation) array of the picture. We just need some way of encoding extension codes and confidence levels as single numbers. To do this, we can convert the confidence into an integer by multiplying it by 1000 and truncating. We add 10000 or 20000 to it if a window at this particular position has been considered as a horizontal or vertical extension respectively and add 30000 if a window at this position has been completely matched. This coding scheme evidently tells us whether a window has already been tried as an extension and if so, from what direction. For example, for a window which has been tried both horizontally and vertically, the entry in the confidence array would be greater than 30000. When we finish region growing, we subtract all such extension codes from the confidence array at positions kept in a record in the process of growing the region.

Note in the above that we record only whether a window was tried as a horizontal or vertical extension, not whether the extension was tried from the left or right or from above or below. This means, for example, that if a window has been tried as a leftward extension of its right neighbouring window, it will not be tried as a rightward extension of its left neighbouring window. Consequently if the window happens to contain two more or less vertical boundaries of the

region of constant displacement, with an intervening textured region of different displacement, a portion of the region of constant displacement may not be tried for merging (see Fig. 4). However, this type of situation occurs so



(a) (b)

Fig. 4. The rectangular object in the first picture (a) has moved rightward relative to the other object in the second picture (b) and only one horizontal extension either from left or from right will not match both edges of the shaded object.

rarely[3] that the increased encoding and decoding cost necessary to distinguish between left and right and between up and down would not be justifiable.

---

3. At least this was the case for the pictures actually used.

4.6 Complete Matching

The question of how to initiate and terminate the processing (matching) of a stereo pair is critical to the speed and accuracy of a matching algorithm. Here by matching we mean finding the displacements of as many edge points[4] as possible. We start processing at the top left hand ccrner of the matchable picture portion. We keep two pointers which give the latest lower bound on the size of the portion already processed. The first is the pointer "VERTIC" which points to the current row, i.e., the row which serves as the uppermost row of the current unmatched window (a window whose match can be found only through global search).

The second pointer "HORIZ" moves horizontally along the current row, such that all pixels to the left of it are considered as the top left hand corner for the window to be matched by exhaustive search. Of course, exhaustive search is not undertaken at all positions. A target area for matching through search is considered according to the

---

4. Since the calculation of correlation thresholds requires autocorrelation estimates based on self-displaced windows, we can process only a portion of an M by N picture. The dimensions of this portion are M-2 by N-2, since we cannot calculate the threshold for the window whose upper left hand ccrner is situated at left hand upper corner of the picture. The same is true for all windows touching any border of the picture.

criteria discussed in chapter 3. For example, search is not carried globally if the number of unmatched edge points is less than half the number of edge points within the window.

Initially, we set both pointers VERTIC and HORIZ to 2. We increment HORIZ by 1 (move to right), each time the window at (VERTIC, HORIZ) is not selected for exhaustive search. When a match is found and extended, chances are that HORIZ will be incremented repeatedly; once it reaches the rightmost pcsition (N-N1+1), we consider the next row. When VERTIC reaches the lowest position (M-M1+1), the processing stops.

In the above approach to terminating the processing, bcth pointers HORIZ and VERTIC move quickly towards the terminal values when large regions of constant displacement are found, and slowly when small objects at different depths are encountered. In other words, the algorithm proceeds ccnservatively for difficult regions (depth highly variable) and more rashly for simple regions (depth uniform over large segments); in the latter case most of the window positions are matched through a small number of exhaustive searches fcllcwed by extensive region-growing.

Chapter 5

Experimental Results and Conclusions

5.1 Comparison with Hannah's Search Method

We have given theoretical reasons for believing that the innovations incorporated into the present matching algorithm permit faster, more accurate, and more complete matching of stereo views.

The key to improved efficiency is the new method of "up and down correlation" for preselection of candidate matching windows. In order to substantiate our claims about the advantages of this method experimentally, we have programmed Hannah's "similarity" technique for preselection as well as the subsequent detailed search using cross correlation in a square neighbourhood of preselected points, and compared the results with our "up and down correlation" method of preselection followed by detailed search in a vertical neighbourhocd of preselected points.

Results were obtained through efficient implementation of the follcwing version of the "Similarity" technique. The

statistical features are the mean and the variance[5] of the two windows to be matched, and similarity (weighted distance between the two vectors) is calculated on the nodes of the grid formed by every mth row and every mth column of the second picture. For the sake of efficiency, we performed all the arithmetic on integers rather than on real numbers. For example, means and variances in two stored arrays were actually the total window sums and square of deviations from the mean. Three different sets of weights were attached to the absolute difference between two means and variances; they were (3, 7), (5, 5), (7, 3). In all the cases the results were almost the same (Hannah also employed a trial-and-error strategy for the selection of weights).

After these comparisons have been made at all nodes of the grid, the K most promising points are chosen and cross correlation is applied in their respective 2m+1 by 2m+1 neighbourhoods. Even in these neighbourhoods further savings are achieved by restricting the number of cross correlations by means of local gridding. The most promising point on this local grid is assumed to be the one with highest correlation. Now the cross correlation is calculated at each pcint in the immediate neighbourhood of the most promising pcint (where the immediate neighbourhood extends to the nearest lines of the local grid). In this way the K best

---

5. We chose these statistical features after a personal communication with Hannah.

candidate match points near each of the K preselected points are found; those whose ccrrelation is below threshold are discarded. In our case m and the gridding factor were respectively 4 and 2, and K was 10, a reasonable choice [Hannah, 1974].

Following are the results of the two techniques applied to several arbitrarily selected windows in two different pictures: For a 150 by 150 picture (Fig. 5a) up and down correlation followed by cross correlation took 2.30 seconds of CPU time on an Amdahl 470 to match 15 different 10 by 10 windows. For 11 of these, the selected candidate windows (i.e., those obtained by local search and whose correlation was above threshold) included the correct match. On the other hand the similarity method took about 3.29 seconds and could only match 6 windows correctly. On another 128 by 128 picture (Fig. 12a) the present method took 1.31 seconds to match 4 windows out of 10 correctly and Hannah's method took 2.32 seconds to match only 3 windows correctly. The reason for finding fewer correct matches here is probably the abundance of depth discontinuities in the second picture, a close-up of a chess board. Because of the relative shift of fcreground and background at a depth edge, corresponding windows containing a depth edge are not necessarily "similar", either in Hannah's sense or in our sense of having nearly identical up and down vectors. We can conclude that the method of up and down correlatiCn provides faster and more reliable preselection of matches than Hannah's

similarity method.

Of course, the fact that the correct match is missed in an exhaustive search for a match of an arbitrarily selected window does not mean that incorrect displacements will be assigned by the complete matching algorithms (either Hannah's or ours). Most windows are matched by region-growing rather than exhaustive search, and region-growing is more likely to succeed around a correctly matched "nucleus" than around an incorrect match. Also our region-growing method includes refinements to avoid mismatches near depth edges.

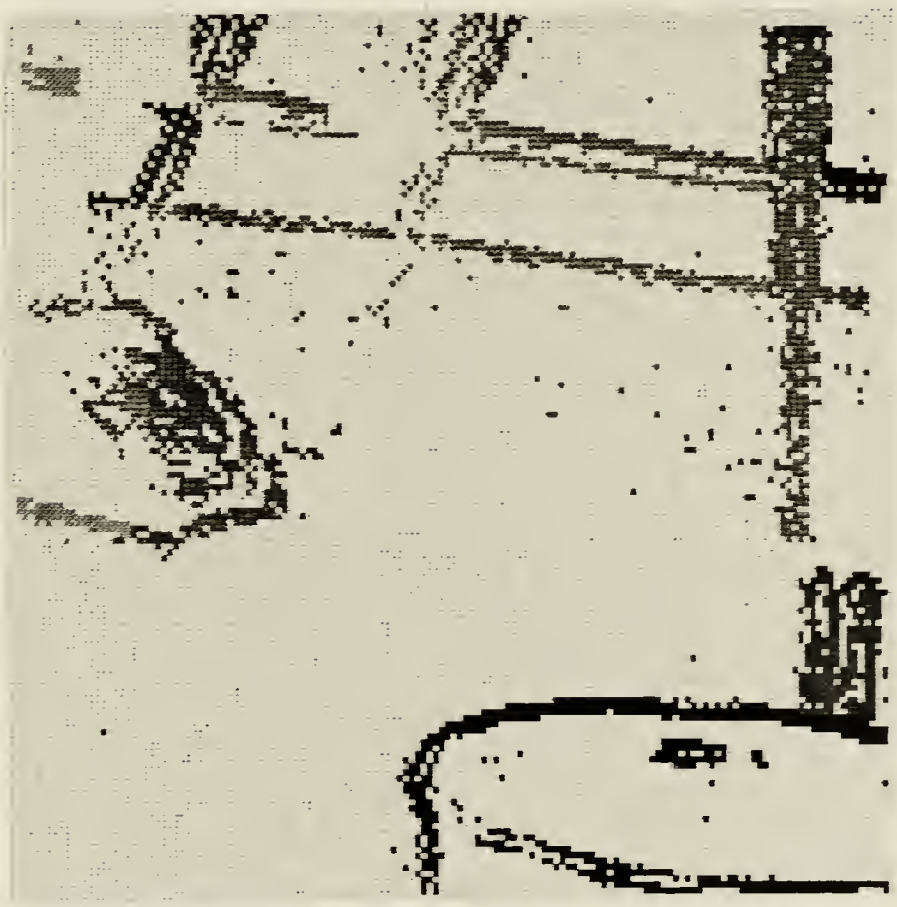
5.2 Results

The complete matching algorithms (with up and down correlation) have been applied to a number of stereo pictures. The output comprises 2 matrices, one for horizontal and one for vertical displacements of edge points in the first picture.

The further an object moves leftward in the second picture relative to its position in the first picture, the closer it must lie to the camera. For the purpose of display, therefore, we have taken the horizontal displacement as the measure of proximity of objects in a scene. Nearby objects are made to appear dark and remote objects lighter by making the maximum horizontal displacement correspond to grey level 0 and the minimum to a

very high one (say 55) (Figs. 5c, 10c, 11c).

The complete matching process took 13.9 seconds on an AMDHAL 470 running under the MTS time-sharing system for a pair of pictures (Figs. 5a,b) of a room scene digitized on a



(c)

Fig. 5. The horizontal displacements of edge points displayed as grey levels.

150 by 150 grid. Virtually all of the more prominent edge points of the image have been assigned correct displacements. Note that part of the telephone is assigned incorrect displacement (Fig. 5c). This problem can occur for any target area whose dominant features are a near-horizontal straight boundary belonging to an object at one distance and an arbitrary shaped boundary of a second object

(a)

(b)

Fig. 5. The left picture of a stereo pair depicting a room scene with telephone, chair, waste-basket and a rectangular block just behind the basket (a). The right picture of the same stereo pair (b).

at a different distance but with the same vertical

displacement (see Fig. 6). The two boundaries need not even

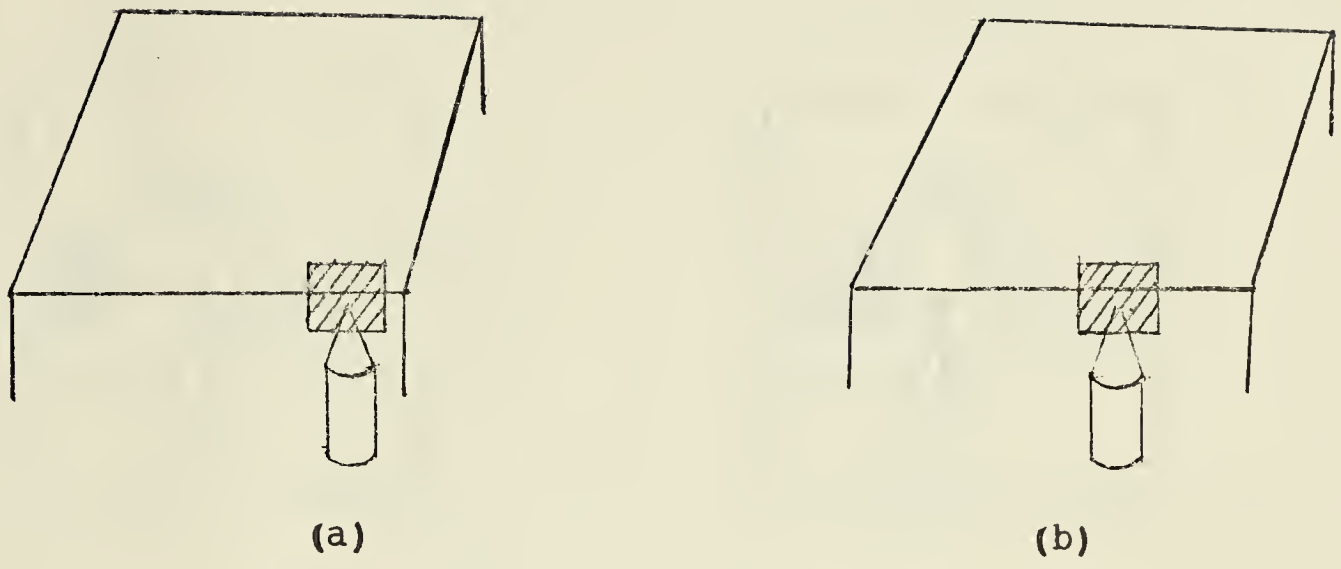


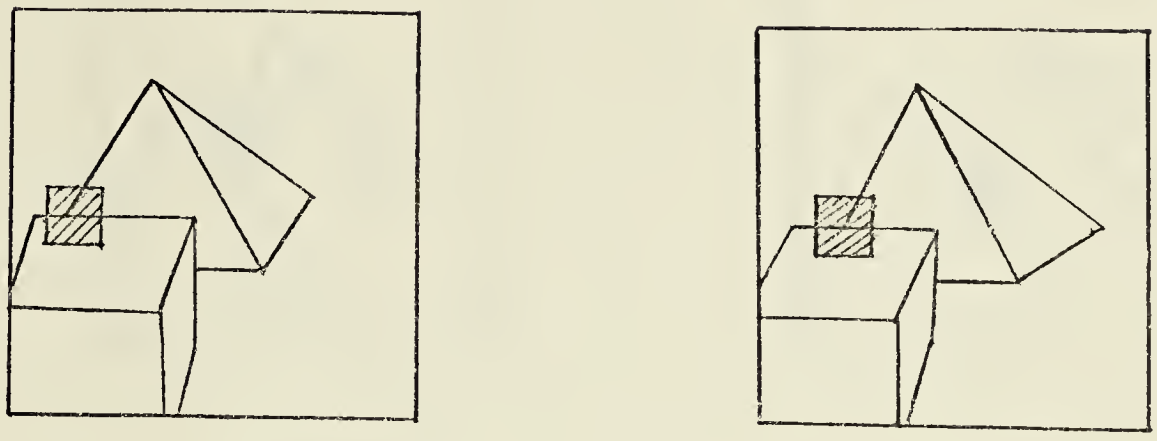
Fig. 6. The shaded window (a) is matched with shaded window (b) which apparently contains the same pattern, but contains a different part of straight edge of the table.

lie within the same window (as happened here) for this error to occur. During region growing, if a region is extended across a near-horizontal boundary, the extension will succeed regardless of the actual horizontal displacement of the cbject relative to the boundary, since we cannot distinguish between parts of a straight edge.

An interesting feature in the output is the correct assignment of displacement on the waste-basket's rim which forms a T-junction with the object in the background. In the original algorithm, this displacement was the same as the object's. This was happening for the following reasons.

For a T-junction, the stem and the cross bar are usually the edges of objects at different depth [Ganapathy, 1975]. Thus if the edge cf the closer object is horizontal, it will be intersected by the edge of the more remote object

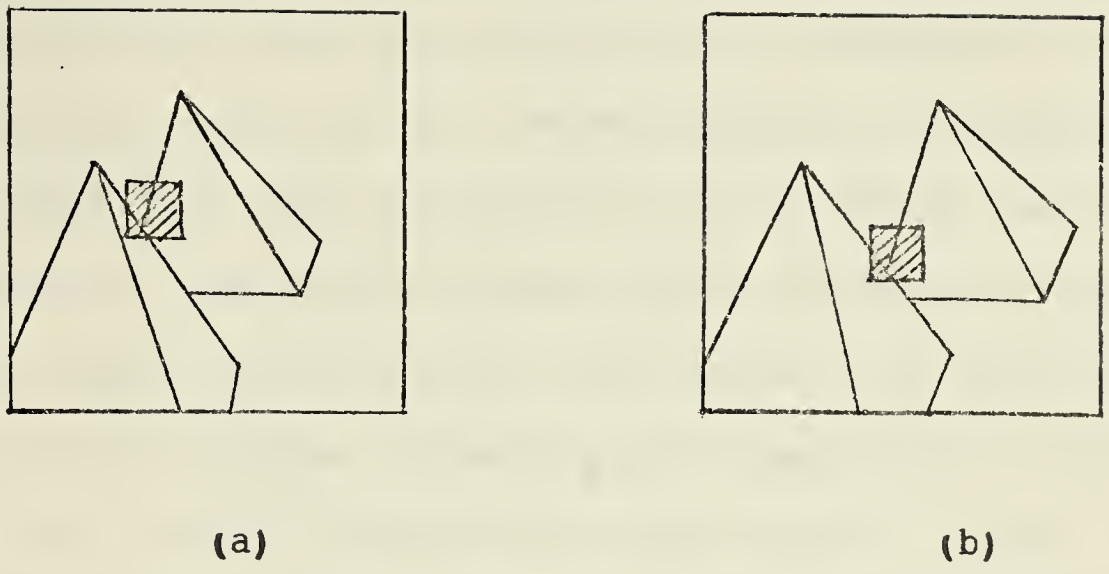
at different points in the two views (see Fig. 7).



(a) (b)

Fig. 7. A match of a T-junction which may cause an erroneous horizontal displacement to be assigned to the closer edge.

Consequently the horizontal displacement of the more remote object will tend to be erroneously assigned to points on the closer edge. If the edge of the closer object is not horizontal, then different parts of the more remote edge will be occluded in the two views (see Fig. 8). In this case incorrect horizontal and vertical displacements will in general be assigned to both edges. The reason that our algorithm often did not succumb to this difficulty lies in the refinements in the region-growing strategy discussed in chapter 4. For example, if we had stopped extending a match upon reaching a uniform region, then the basket's rim which occludes the object in the background would not have been correctly matched, since it is only during an upward extension from the uniform interior of the basket that the



(a) (b)

Fig. 8. A match of a T-junction which may cause erroneous horizontal and vertical displacements to be assigned to both the closer and the more remote edge.

rim is attributed the same displacement as the basket. Also, the match of the rim would still have been missed without the technique of backing up a few pixels upon reaching the limits of a region of constant displacement (discussed further below).

Another small region which is falsely matched is the baseboard's rightmost portion. The reason for this problem and a possible approach (not implemented) for dealing with it are as follows.

When the match for a window is found, it may be that the corresponding region or a part of it in the second picture has already been matched with a different region of the first picture. But, of course, any region in the second picture can correspond to at most one region in the first picture. The above situation can occur when two objects or

their parts look similar and one of these objects is occluded partly or entirely in the second picture. The area which is occluded will be matched with the second object's ccrresponding region in the second picure. An approach to dealing with this type of mismatch is to check the sizes of the regions in the first picture whose corresponding regions overlap in the second picture. The measure of this size can be taken as the number of edge points matched in each of the regions. We discard the match of the smaller region. This criterion is based on the intuition that only the occluded part (hence probably the smaller in size) is likely to be mismatched. To implement this criterion, we need to keep four more arrays. One array is a logical array which points to the edge points in the first picture which are incorrectly matched and should be taken as unmatched in the final output displacement matrices. Two arrays are needed for pointing to these edge points at the stage when we are in an overlapping region in the second picture, and one for keeping count of the number of edge points whose corresponding areas are reliable (under this criterion). The region extension algorithm also needs to be modified, since we can no longer assign the displacements to edge points unless we know its extended size (a measure of reliability). Thus, we can divide our region growing algorithm into two phases: (1) extension phase, (2) assignment phase. After growing a region and counting the total number of edge points in it, we can assign the displacement to those edge

pcints whose corresponding points in the second picture are not yet matched or matched with a smaller region in the first picture. Of course, we need to save the corresponding correlations, averages, etc., for all matched window pcsitions tc avoid recomputation in the assignment phase.

We also applied the matching algorithms to a pair of pictures of another room scene digitized on a 256 by 256 grid (for left picture see Fig. 9). But we could not get

Fig. 9. The left picture of a stereo pair of another room scene.

substantial matches in a reasonable time. When we closely inspected the picture, we saw that some noise in the form of a ripple was superimposed on the picture. The wave length of this ripple was 2 pixels in the vertical direction. This

noise caused normalized cross correlations to remain below threshold. To overcome this problem, we blurred the original picture by averaging over 2 by 2 windows (Figs. 10a,b), and

(c)

Fig. 10. The horizontal displacements of edge points displayed as grey levels in the blurred stereo pair.

were able to match in about 31 seconds. Note that the rear leg of the chair is matched with its front leg, since the former is occluded in the second picture. The reason for this type of error has already been discussed. The latter pair of pictures was also collapsed into a pair of 128 by 128 pictures (Figs. 11a,b) by replacing 2 by 2 regions by their averages. This pair was matched in 11.5 seconds.

In our original matching algorithm, false assignments

(a)

(b)

Fig. 10. Prior to matching, the original picture of Fig. 9 has been blurred by averaging over 2 by 2 regions (a). The right picture of the blurred stereo pair (b).

of displacements occurred quite frequently. First of all, the use of displacements obtained with highest correlation

(a)

(b)

Fig. 11. Prior to matching, the original picture of Fig. 9 has been collapsed from a 256 by 256 matrix into a 128 by 128 matrix (a). The right picture of the collapsed pair (b).

eliminates many false matches. The matches in Figs. 6,7,8 although they appear to be perfect as drawn, will in fact

(c)

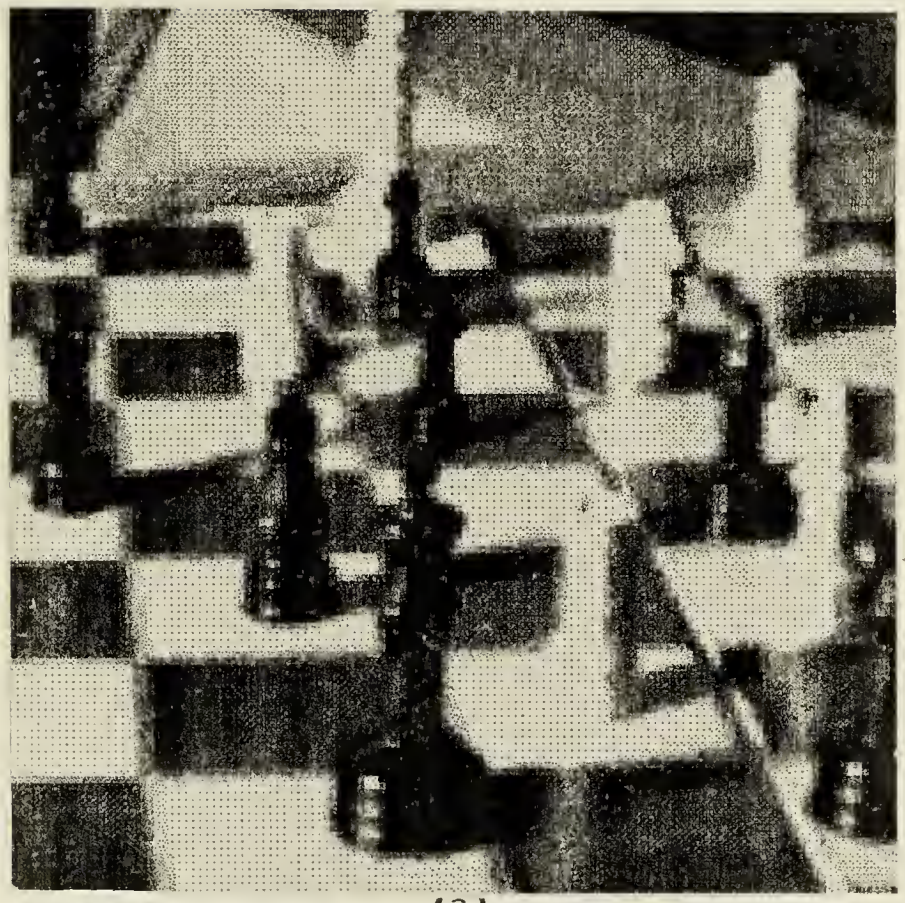
Fig. 11. The horizontal displacements of edge points displayed as grey levels in the collapsed stereo pair.

have non-unit correlation since different portions of an edge are being matched and all real edges have small irregularities of grey level and shape. Thus windows slightly displaced from those shown but containing a boundary of one object only will usually be matched with higher correlation than the erroneous matches. As a result correct displacements will be assigned. However, not all window positions are tried, since we attempt to proceed in full-window steps during region growing, half window steps if this fails and so on. Consequently all window positions tried in the vicinity of the erroneous match might contain

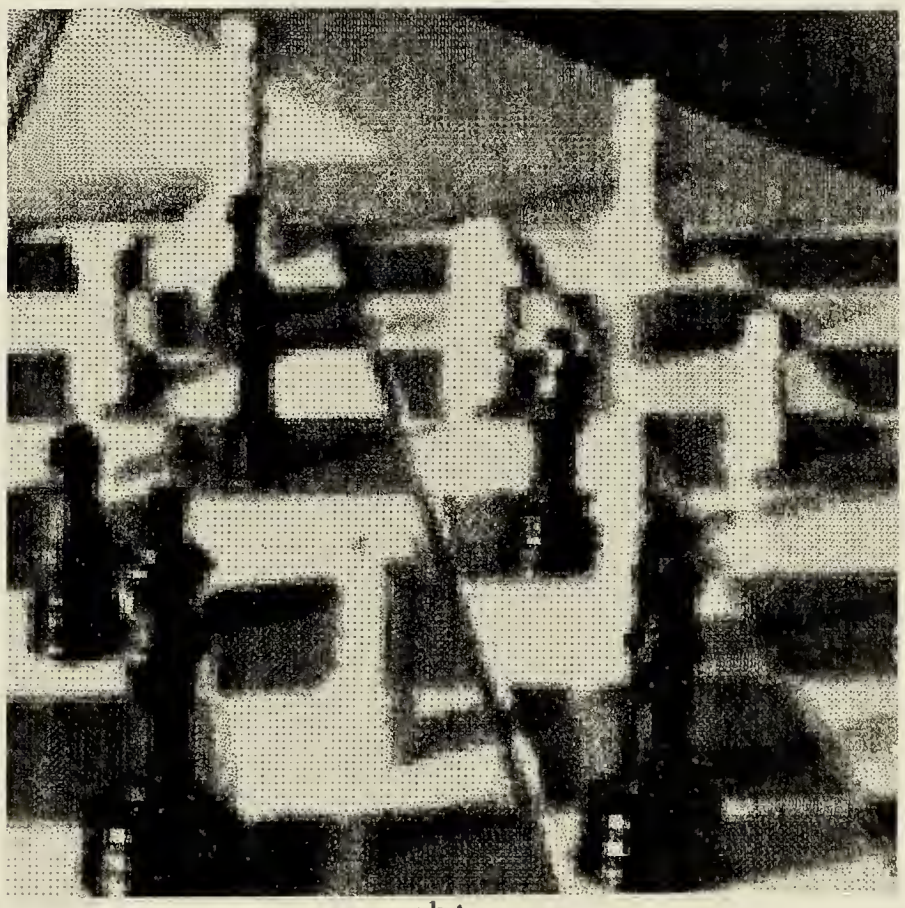
portions of the boundaries of both objects and give correlations no better than that of the erroneous match. This difficulty motivated the refinement of the region growing strategy described in chapter 4, in which a window is "backed up" one pixel at a time for a few steps when the window can be advanced no further in a particular direction. As pointed out, this refinement tends to eliminate assignment errors near boundaries between objects. For example, the rim of the waste-paper basket forming a T-junction with the object in the foreground of Figs. 5a,b was assigned correct displacement only after the introduction of the refinement in region-growing strategy.

The matching algorithms were applied to a close-up of a chess board (Figs. 12a,b). The lateral distance between two camera positions was about 3/2 inch and the distance of the board from the lens about 18 inches. Fig. 12c gives a rough idea of relative depths of different pieces. Most of the picture has been correctly matched. Some board square contours near the left hand edge of the picture are not visible in the second picture; but due to the presence of similar contour patterns on the board the mismatch has taken place. A small part of the black bishop is also mismatched. The receding lines near the top edge of the picture are mismatched because of the T-junction problem. A portion near the bottom right hand corner is also mismatched.

Fig. 13c depicts relative depths of a close-up of another chess board position. In this pair (Figs. 13a,b) the



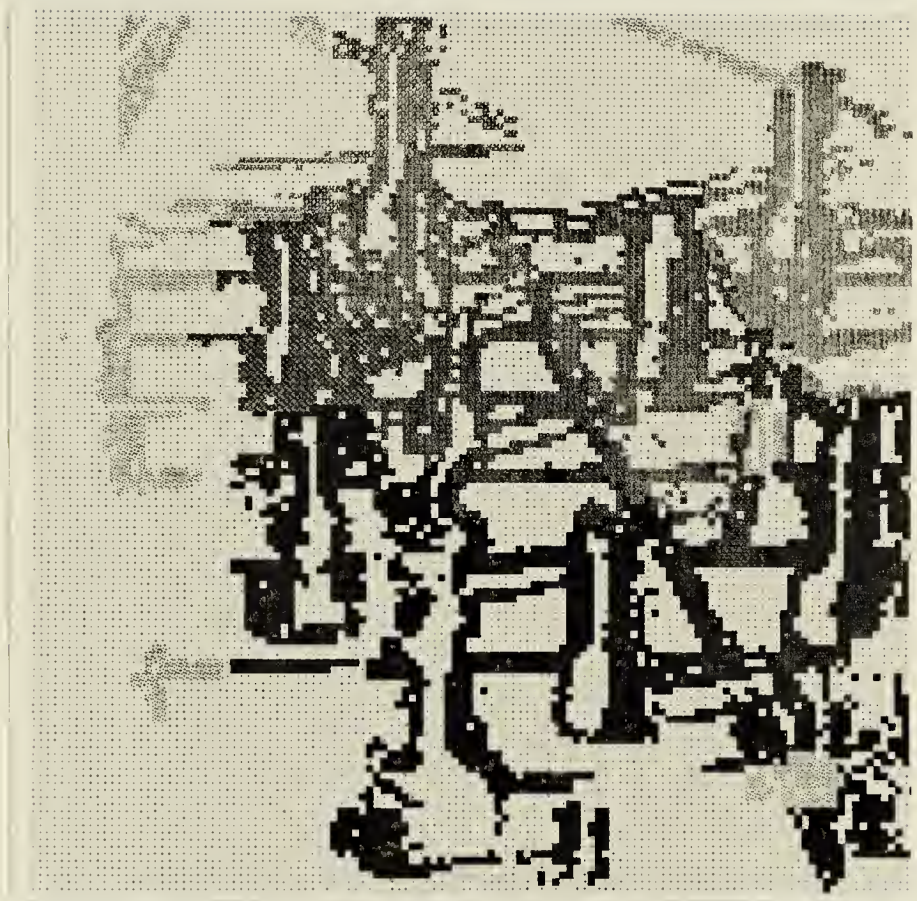
(a)



(b)

Fig. 12. A close-up of a chess board position (a). The right picture of the same position (b).

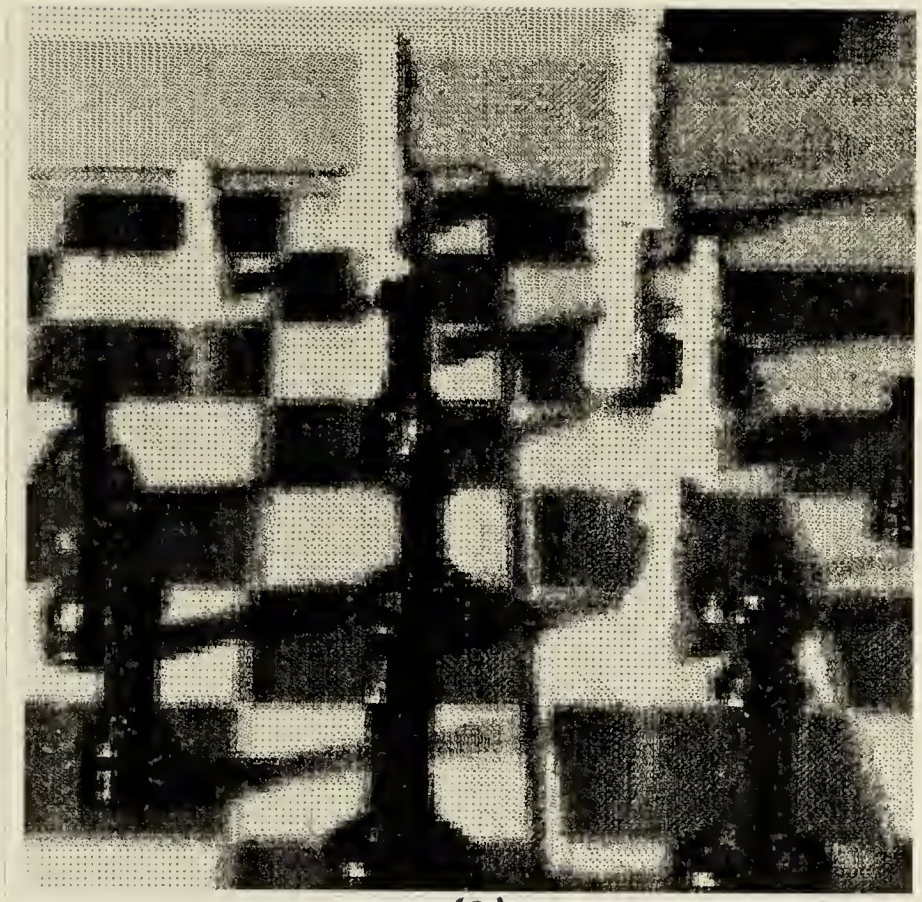
camera's displacement was about 3/2 inch vertically downwards. So the measure of proximity should naturally be the vertical displacement. In matching this pair again some



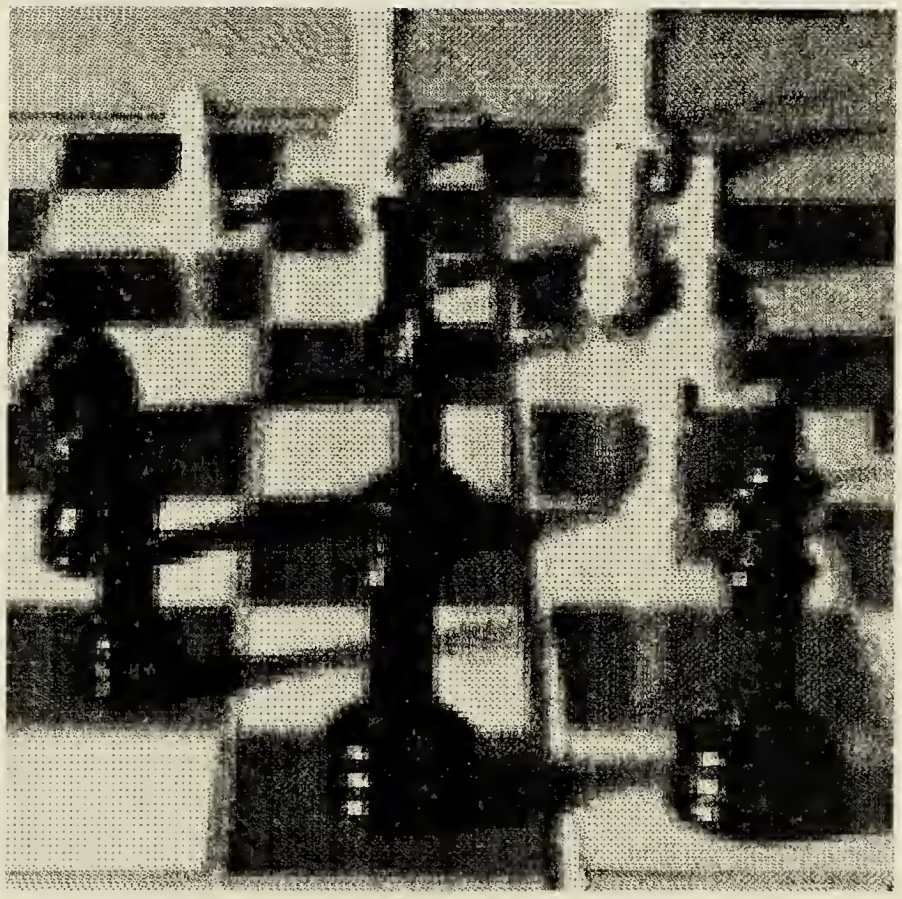
(c)

Fig. 12. Horizontal displacements of edge points displayed as grey levels.

small areas are mismatched. The top parts of white king and queen are not visible in the seond picture but are accidently matched with their respective lower parts. The slightly darker spots in the light region in the upper part of the picture are not errors, but only "rcund-off noise"; i.e., they correspond to 1-pixel changes in displacement. Such changes are to be expected since displacement is



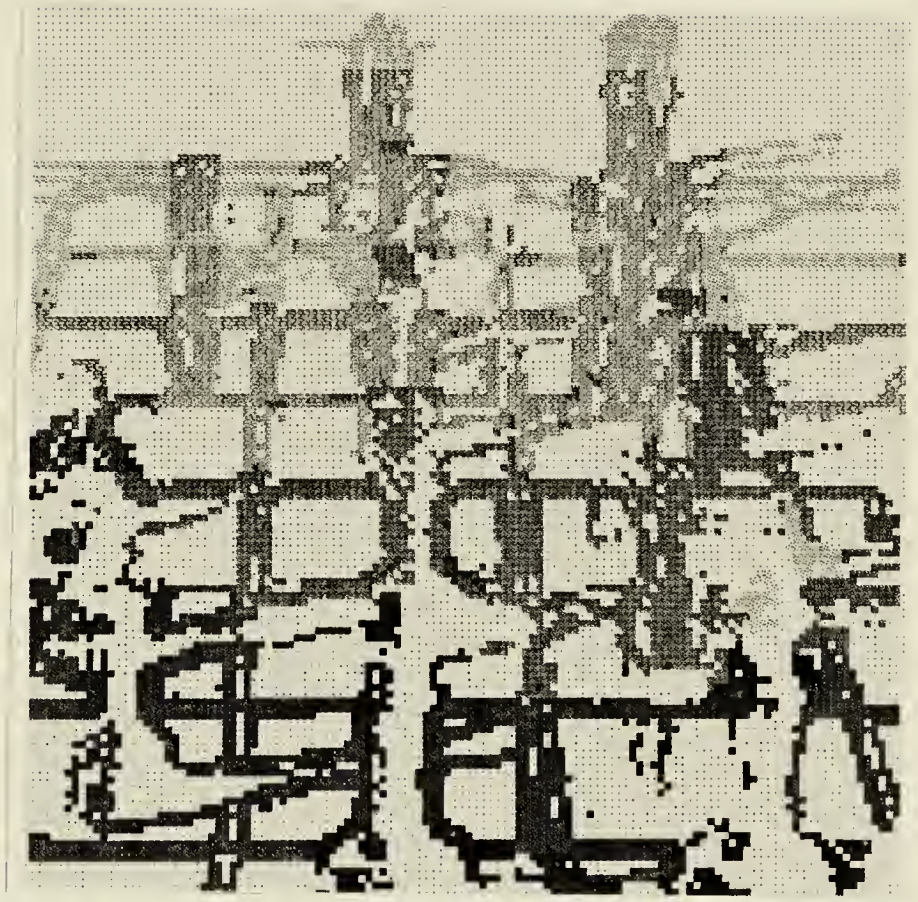
(a)



(b)

Fig. 13. A close-up of a different chess board position (a). A close-up of the same position after moving the camera vertically downward by about 3/2 inch (b).

measured only to the nearest number of pixels. Also the upper part of the black king (near right hand edge of the



(c)

Fig. 13. Vertical displacements of edge points displayed as grey levels.

picture) is mismatched.

5.3 Conclusions

The achievements of the current project lie in the fcllcwing facts. The selection criteria based on grey level changes are less dependent on grey level scale than criteria based on variance. The line finder subroutine doesn't involve the determination of the autocorrelation surface in the neighbourhood of a target area. Instead, it makes a decision on the basis of edge points' spatial distribution in the window and, therefore, is much more efficient. However, it is incapable of detecting multiple parallel edges. In searching, the closeness measure up and down correlation for a 10 by 10 window in the majority of cases is about 140 times faster than the computation of normalized cross correlation. Moreover, up and down correlation provides a method of preselection which is insensitive to changes in average grey level and contrast between the two views. The method compares favourably with respect to both speed and accuracy with Hannah's method of preselection. The displacement assignment method is more accurate than previous criteria, since it limits assignments to edge pcints contributing positively to the cross correlation. Further, it allows an edge point to be matched with more than one displacement and chooses the most reliable of these. The modification of the standard region growing algorithm introduces a saving factor of one of the dimensions of the target window, and its refinement makes it possible to match edge points near depth edges with greater

accuracy.

While some mismatches still occur with our method, the same is undoubtedly true for previously proposed methods. To quote Thompson [1975]:

> 'Stereo matching is not an infallible means of analysis. "False matches" are an ever-present problem arising for two reasons. There may be multiple, highly correlating matches caused by repetition of features: imagine trying to match two views of a freshly-painted picket fence against a uniform background. Also, the discovery of a correct match may be blocked by the occlusion of the scenery seen in one picture by a closer object; a spuriously high correlating point-pair may be selected instead.'

The manner in which results have been presented by previous authors make even qualitative judgements about the reliability of their matching methods very difficult. For example, Hannah [1974] presents only correspondences for selected windows. Levine [1973] shows a sequence of depth contcurs over pictures of rocky terrain; these contours often meander erratically, even in seemingly "flat" regions, so that it is not clear whether the terrain topography has been faithfully mapped. Our theoretical reason for assuming that previous methods are at least as error-prone as ours is simply this: these methods use the same ultimate criterion for selection of the best match among preselected candidates as we do, namely the value of the normalized cross correlation. Furthermore, they employ the questionable criterion of attributing the displacement of a matched window to the centre of the window.

5.4 Problems for Further Investigation

As seen, a T-junction normally causes mismatches of at least one of the edges involved in it. Detection of T's is a simple task in line drawings of polyhedral objects but not in digital images of real scenes. Proposing criteria which can get rid of mismatches due to T-junctions will be a useful improvement in the current approach.

We have proposed one criterion for not allowing any region in the second picture to be matched with more than one area in the first picture, but we are not sure whether this will work in general. This is another interesting problem worthy of further investigation.

The addition of programs to obtain depths from displacements (given camera parameters) should be straightforward.

The next major step required to achieve computer stereo vision is the development of programs to interpret the matched images, so as to produce a description of the depicted scene in terms of physical objects and relationships between them. This is a familiar and difficult problem in artificial intelligence (rather than picture processing), since meaningful descriptions can only be produced with the aid of "world knowledge" to guide the interpretion process.

Appendix

Matching in Pidgin Algol

```
procedure TMATCH(M,N,M1,N1,LPICT,RPICT,EDGE)
begin
comment LPICT(i,j) and RPICT(i,j) (1≤i≤M; 1≤j≤N) are
 respectively the left and right hand pictures of
 the stereo pair. The picture LPICT is named as the
 first picture (i.e., containing the windows for
 which matching windows are sought). The picture
 RPICT is named as the second picture (i.e., the one
 searched for matches). EDGE(i,j) represents the
 edge code matrix. All the windows whose matches are
 sought are M1 by N1. The X-component of the
 displacement of the pixel LPICT(i,j) is DISPX(i,j)
 and the Y-component is DISPY(i,j), and CONF(i,j) is
 the latest value of its correlation confidence for
 with which it was last matched;
IBMM1:=M-M1;
NMI1:=N-1;
JRNN1:=N-N1;
comment The IBMM1th row is the last row which can
 serve as the uppermost row of a window to be
 matched. The JRNN1th column is again the last
```

```
          column which serves as the left most column of a
          window to be matched. Window positions including
          border pixels are not used, since it must be
          possible to consider one pixel shifs for each
          position for calculating the autocorrealtion
          threshold;
        for J:=1 until do
          UMEDGE(J):=sum of unmatched edge points in the Jth
          column of width equal to the height of a window,
          the first pixel is at VERTIC;
        VERTIC:=2;
start:  HORIZ:=2;
        Place a window with its top left hand corner at
          (VERTIC, HORIZ), sum all the edge points determined
          by pairs of pixels within the window, call it
          IESUM;
        if IESUM>10 then goto check1;
mcve:   HORIZ:=HORIZ+1;
        if HORIZ>JRNN1 then do
           begin
           VERTIC:=VERTIC+1;
           goto check
           end;
check1: Calculate IESUM2, the half of the total number of
          edge points in the window;
        Let IUMEDG be total number of unmatched edge points
          in the window;
```

```
        if IUMEDG<IESUM2 then goto move;
        comment Test, if there is only one straight edge
         through the window;
        if LINE(VERTIC,HORIZ)=true then do
           begin
           comment Test, whether the edge is broken;
           if BROKEN(VERTIC,HORIZ) then goto search
           else goto move
           end;
search: MATCH(VERTIC,HORIZ,ID,JD)
        comment (ID,JD) are respectively the X-component and
         Y-component of of displacement of a target window
         in left picture relative to the corresponding
         window in the right picture, if a unique match was
         found;
        if a unique match was found then do
           begin
           Assign the displacement (ID,JD) to all edge
            points in the window, save those edge points
            which don't contribute positively to the
            normalized cross-correlation and those which are
            already matched through other window position
            with a higher correlation;
           comment Then region of constant displacement is
            grown;
           EXTEND(VERTIC,HORIZ,ID,JD);
           Update the number of unmatched edge points in the
```

```
            columns in the matched window at (VERTIC,HORIZ)
           end;
        goto move;
check:  if VERTIC>IBMM1 then goto exit;
        for j:=2 until NMI1 do
        Update UMEDGE(j) for each column;
        goto start;
exit:   Output the displacement matrices
        end.


        logical procedure LINE(VERTIC,HORIZ)
        begin
        comment This procedure tells whether all the edge
         points except a few (say NUMBER) of the M1 by N1
         window at (VERTIC,HORIZ) lie along one straight
         edge through the window. If so, LINE is set to true
         , otherwise false;
        Let (XG,YG) be the center of gravity of all edge
         points (say L) in the window;
        Let XSQ be the sum of squares of X-coordinates and
         XYP be the sum cf products of X and Y-coordinates
         of all edge points;
        comment If the fitted line is represented by
         Y=a+b*X, then calculations for a and b are done
         through least square techniques;
        S1:= XYP-L*XG*YG;
```

```
if S1=0 then do
   begin
   comment There is only one edge, and it is
   vertical;
   if the edge doesn't end in the window then do
      begin
      LINE:= true;
      for I:=1 until M1 do
      if Ith row is cut by the edge then ROW(I):=1
      else ROW(I):=0
      end
   else LINE:= false;
   goto exit;
   end;
S2:=XSQ-XG*XG;
comment If the following test passes, then there is
 only one straight edge, and it is horizontal;
if S2=0 then do
if the line doesn't end in the window then do
   begin
   LINE:=true;
   for J:=1 until N1 do
   if Jth column of the window is cut by the edge
   then COL(J):=1 else COL(J):=0;
   end
else LINE:= false;
goto exit;
```

```
b:= S1/S2;
a:= YG-b*XG;
if ABS(b)<1 then do
   begin
   comment Take the deviation of edge points from
    the fitted edge in terms of their vertical
    distances from it;
   Let TOTAL be the number of edge points in the
    window lying D or more pixels away from the
    edge;
   for I:=1 until M1 do
   if Ith row of the window is cut by the edge then
    ROW(I):=1 else ROW(I):=0;
   if TOTAL>NUMBER then do
      begin
      LINE:=false;
      goto exit;
      end
   else if the line passes through both the ends of
   the window then LINE:=true else LINE:= false
   goto exit
   end
else do
   begin
   comment Take the deviation of edge points from
    the fitted edge in terms of their horizontal
    distances from it;
```

```
Let TOTAL be the number of edge points in the
 window lying D or more pixels away from the
 edge;
for J:=1 until N1 do
if Jth column of the window is cut by the edge
 then COL(J):=1 else COL(J):=0;
if TOTAL>NUMBER then do
   begin
   LINE:=false;
   goto exit;
   end
else if the line passes through both the ends of
the window then LINE:=true else LINE:= false
end
exit:  end.
```

```
logical procedure BROKEN(VERTIC,HORIZ)
begin
comment This subroutine tells whether a straight
 edge passing through the window at (VERTIC,HORIZ)
 is broken or not. If so, BROKEN is set to true,
 otherwise to false, depending on the slope of the
 edge, it has already been observed which columns or
 rows are intersected by the edge in the LINE
 procedure;
The slope of the edge is already calculated in the
```

```
 subroutine LINE;
if abs(slope)<1 then do
   begin
   comment For an edge to be broken, there has to be
    a gap of at least two rows of the window which
    are not intersected by the edge, and on either
    side of this gap there should be at least three
    consecutive rows intersected by the edge;
   if the edge is broken then BROKEN:= true
   else BROKEN:=false;
   end
else do
   begin
   comment In this case for an edge to be broken,
    there has to be a gap of of at least two columns
    in the window which are not intersected by the
    edge and on either side of this gap there should
    be at least three consecutive coumns which are
    intersected by the edge;
   if the edge is broken then BROKEN:= true
   else BROKEN:=false
   end
end.
```

```
procedure MATCH(IX,JY,ID,JD)
begin
comment (IX,JY) is the top left hand corner of the
 target window at (IX,JY) and (ID,JD) is its
 displacement. GEXACT, FOUND are logical variables.
 GEXACT is true if a reliable match has already been
 found. FOUND is true as long as nc ambiguous match
 is found for the current window. RUDV(I,J),-
 ...,RUDV(I,J+N1-2) represents the up and down
 vector for the window at (I,J) in the second
 picture. HORSTA, HOREND, VERSTA, VEREND are
 respectively the horizontal and vertical search
 bounds calculated on the basis of maximum absolute
 and relative displacements;
      if GEXACT=true then goto 1;
      Set the search bounds on the basis of maximum
       possible absolute displacement possible;
      goto 2;
1:    Set the search bounds on the basis of maximum
       possible relative displacement;
2:    FOUND:=true;
      comment ITIMES is the number of found unambiguous
       matches and CORMAX is the highest correlation among
       the accptable correelations;
      ITIMES:=0;
      CORMAX:=0;
      M11:=M1-1
```

```
N11:=N1-1;
N111:=N1-2;
comment Determine UDV, IUDTH (up and down threshold)
 for the target area and ISKIP (the skip factor in
 vertical direction for UDV matching);
UDTHLD(IX,JY,ISKIP,IUDTH,UDV);
comment Calculate TH (autocorrelation threshold);
THRCLD(IX,JY,TH);
if TH<.5 then goto fail;
if IUDTH =0 then goto 4;
comment Compare the UDV's element by element;
for J:= HORSTA until HOREND do
   begin
for I:=VERSTA until VEREND step ISKIP do
   begin
   for K:= 1 until N11 do
   if the Kth element of two UDV's don't match
    then goto loop2;
   comment A promising area has been found;
   IHORI:=J;
   IVERTI:=I;
   comment See if any area in the vicinity of
   promising area passes the normalized cross
   correlation;
    FINCOR(IX,JY,IVERTI,IHORI,ISKIP,COR,ID,JD);
    if FOUND=true then goto loop2;
    comment An ambigous match has been found;
```

```
                goto fail;
loop2:          end
             end;
          comment Search for the target area is complete;
          goto check;
          comment Up and down threshold is too liberal;
3:        if IUDTH> N111 then goto fail;
          IFROM:= IUDTH+1;
          for J:=HORSTA until HOREND do
          for I:=VERSTA until VEREND step ISKIP do
             begin
             Let RELSUM be the sum of absolute differences of
           first IUDTH corresponding UDVs' elements;
          for K1:=IFROM until N11 do
             begin
             Let K2 be the absoluted difference of the
           K1th correspondong elements of two UDV's;
             comment No match through up and down
           correlation;
             if RELSUM>IUDTH then goto loop1;
             end;
          comment A promising area has been found;
          IHORI:=J;
          IVERTI:=I;
          FINCOR(IX,JY,IVERTI,IHORI,ISKIP,COR,ID,JD);
          if FOUND=true then goto loop1;
          goto fail;
```

```
loop1:      end;
check:   if ITIMES<1 then goto fail;
         comment An unambiguous match has been found;
         goto stop;
fail:    Either the target area is not suitable or ambiguous
         match has been fcund or not a single match has been
         found
stop:    end.

         procedure UDTHLD(IX,JY,ISKIP,IUDTH,UDV)
         begin
         IXM1:=IX+M11;
         JYN1:=JY+N11;
         comment (PIXSUM(J), J=JY,JYN1) are the sums of M1
          pixels of the Jth column. These sums are used to
          calculate UDV's. (PSQSUM(J), J=JY,JYN1) are the
          sums of square of M1 pixels of the Jth column PSUM
          and PSSUM are respectively the sum and squares of
          all pixles in the target window and are passed to
          the subroutine THROLD;
         comment Calculate UDV for the target window at
          (IX,JY);
         JYN11:=JYN1-1;
         J1:=JY-1;
         for J:=JY until JYN11 do
            begin
```

```
      K1:=PIXSUM(J)-PIXSUM(J+1);
      if K1>0 then UDV(J-J1):=2
      else if K1=0 then UDV(J-J1):=1
      else UDV(J-J1):=0;
      end;
comment Determine IUDTH and ISKIP;
Move the target window vertically upwards and
 downwards by 1 pixel and calculate UDV's in the new
 positions. IUDTH is the smaller of the sums of
 absolute differences of the corresponding elements
 of new UDV's from the undisplaced window's UDV.
 Keep on moving the window in one or both directions
 as long as this difference sum doesn't cross IUDTH.
 The distance travelled in both direction is the
 skip factor;
exit:  end.

      procedure FINCOR(IX,JY,VERTIC,IHORI,ISKIP,-
      CORMAX,ID,JD)
      begin
      comment {(XMATCH(I),YMATCH(I)),I=1,45)} are the
       coordinates of the top left hand corner of the
       found matches. In order not have any ambiguous
       match (i.e., not seprated by more than two pixels
       in either direction), the number of matches
       shouldn't exceed 45. CORMAX is the maximum
```

```
 correlation among the found matches. (IXMAX,JYMAX)
 is the position to which CORMAX corresponds.
 VERSTA, etc., are the same values as in subroutine
 MATCH;
comment Set the vertical and horizontal search
 bounds through normalized cross correlation;
IVERI:=VERTIC-ISKIP;
if IVERI<VERSTA then IVERI:=VERSTA;
IVERT:=VERTIC+ISKIP;
if IVERT>VEREND then IVERT:=VEREND;
IH1:=IHORI-1;
IH2:=IHORI+1;
if IH1<HORSTA then IH1:=HORSTA;
if IH2>HOREND then IH2:=HOREND;
for I:=IVERTI until IVERSK do
for J:=IH1 until IH2 do
   begin
   Let COR be the normalized cross correlation
    between target window and window situated at
    (I,J) in the second picture;
   if COR≤TH then goto loop;
   ITIMES:=ITIMES+1;
   if ITIMES>45 then goto 1;
   if COR≤CORMAX then goto 2;
   CORMAX:=COR;
   Calculate ILAV and IRAV the averages of grey
    levels in the target and the corresponding
```

```
            window for the displacement assignment purposes;
            IXMAX:=I;
            JYMAX:=J;
            ID:=I-IX;
            JD:=J-JY;
2:          IXMATCH(ITIMES):=I;
            JYMATCH(ITIMES):=J;
            comment Check for the ambiguity of the match;
            for K:=1 until ITIMES do
            if IABS(IXMATCH(K)-IXMATCH(ITIMES))>2 then goto 1
            else if IABS(JYMATCH(K)-JYMATCH(ITIMES))>2 then
            goto 1
1cop:       end;
            goto exit;
1:          FOUND:=false;
            comment An ambiguous match has been found
exit:       end.

            procedure EXTEND(IX,JY,ID,JD,EXACT)
            begin
            comment (IX,JY) are the coordinates of the top left
             hand corner of a matched window with displacement
             (ID,JD). The region with displacement (ID,JD) is
             grown around this window. GEXACT is a logical
             variable which, once becomes true, remains true
             throughout the processing. GEXACT= false means that
```

a reliable match has not yet been found. LEXACT is another logical variable which becomes true when IEXT exceeds 5 (ie. the match has been extended by at least 5 full windows). This means that the current match is reliable and we can use its displacement to adjust the search bounds. ITOP represents the length of a stack which stores the positions of the windows around which the current region can be grown. These positions are stored in two arrays ISEED and JSEED;

```
        ITOP:=1;
        LEXACT:= false;
        IEXT:=IEXT+1;
        ISEED(ITOP):=IX;
        JSEED(ITOP):=JY;
        if GEXACT= false then do
           begin
grow1:     STACK(ID,JD,ITOP,ISEED,JSEED);
           comment Stop growing when the match is not
            extendable by a full window in at least one of
            the four directions and the length of the stack
            reaches 1;
           if ITOP<2 then goto exit;
           if ITOP>5 then do
              begin
              comment A reliable match has been found;
              GEXACT:= true;
```

```
            LEXACT:= true;
            goto grow1
            end
        else do
           begin
grow2:     STACK(IX,JY,ID,JD,ISEED,JSEED);
           if ITOP<2 then goto exit;
           goto grow2
           end;
exit:   if IEXT>5 then LEXACT:= true;
reset:  Subtract extension codes from confidences of all
         pixels at which windows were placed in the first
         iteration for extending the region;
        end.
```

```
procedure STACK(ID,JD,ITOP,ISEED,JSEED)
begin
comment A region of constant displacement (ID,JD) is
 grcwn in the immediate neighbourhood (up, down,
 left, right) of the matched window at (IX,JY);
IX:=ISEED(ITOP);
JY:=JSEED(ITOP);
comment ITIMES is a variable which is incremented by
 1 whenever it is not possible to extend the region
 by a full window in the direction of growth;
ITIMES:=0;
```

```
        for I:=1 until 4 do
           begin
           if i=1 then goto down;
           if i=2 then goto up;
           if i=3 then goto right;
           comment Try to extend the region to the left;
left:      if an attempt has already been made to extend the
            match horizontally to the left of the window at
            (IX,JY) then do
              begin
              ITIMES:=ITIMES+1;
              goto stop
              end
           else do
              begin
              comment JYIth column is the first of JW
               columns (maximum N1 in one extension) which
               can possibly be merged and each is of M1
               pixels length starting from IXLth row of the
               first picture;
              IXL:=IX;
              JYL:=JY-1;
              LEBIN(IXL,JYL,ID,JD,JW);
              if JW<N1 then do
                 begin
                 comment Extension by a full window is not
                  possible;
```

```
            ITIMES:=ITIMES+1;
            Add the horizontal extension code to the
             correlation confidence of the pixel at
             which the window was placed in the first
             iteration of the extension procedure;
            goto assign
            end
          else do
            begin
            comment Extension by a full window is
             possible;
            IEXT:=IEXT+1;
            ITOP:=ITOP+1;
            ISEED(ITOP):=IX;
                    JSEED(ITOP):=JY-N11;
                    goto assign;
                    end;
dcwn:   Analogous to left in the downward direction;
        goto assign;
right:  Analogous to left in the rightward direction;
        goto assign;
up:     Analogous to left in the upward direction;
assign: Assign the displacement (ID,JD) to all edge points
         in the newly merged portion, save those which do
         not contribute positive to the correlation and
         those edge points which are already matched at
         other displacement with higher correlation
```

```
          confidence;
check:    comment If no full match is possible in any of the
          four directions full match is not possible,
          decrease the length of the stack by 1;
          if ITIMES=4 then ITOP:=ITOP-1;
stop:     end.


          procedure DOBIN(IX,JY,ID,JD,IW)
          begin
          comment This subroutine tries to extend the current
          region with displacement (ID,JD) vertically
          downwards starting from IXth row (the one
          immediately below the matched portion). The
          horizontal bounds for the extension window are (JY,
          JY+N11). ISTA points to upper most row of the
          window in the first picture which is being
          correlated with the corresponding window in the
          secon picture in the current iteration;
          if it has been already tried to extend the region
          vertically by placing a window at (IX, JY) then
          goto stop;
          IW1 (the width by which match can be extended is the
          shift downwards which keeps the target window
          within the first picture and the corresponding
          window in the second picture;
          if IW1≥M1 then IW1:=M1;
```

```
        comment The maximum number of rows which can be
         merged in a single call of this routine is M1;
        ISTA:= IX-(M1-IW1);
lcop:   Place the window at (ISTA,JY), calculate the
         autocorrelation threshold and the correlation
         between the window at (ISTA, JY) in the first
         picture and the window at (ISTA+ID, JY+JD) in the
         second picture;
        if correlation>threshold then do
           begin
           if this is the first iteration (ie. all IW1 rows
            are mergeable in the current region) then goto
            ex1;
           Move the windcw downward by a step I2, I2 is half
            the vertical width of remaining unmatched
            portion out of IW1;
           if I2=0 then goto ex2;
           ISTA:=ISTA+I2;
           goto loop
           end
        else
           begin
           Move the windcw upward by a step I2, I2 is half
            of the width by which the extension was tried in
            the last iteration;
           if the window has reached a position where it was
            already merged in the current region then goto
```

```
            ex3;
            ISTA:=ISTA-I2;
            goto loop
            end;
ex1:    IW:=IW1;
        gotc stop;
ex2:    IW:=ISTA-IX+M1;
        goto back;
ex3:    IW:=ISTA-IX+M1-1;
        ISTA:=ISTA-1;
        comment ISTA is the uppermost row for a window
         position beyond which the match cculd not be
         extended vertically downwards;
        comment This match has not been extended by a full
```

window in the current call of extension routine. This must bebecause of the presence of a depth edge. In order to increase the reliability of displacement on the pixels near this part of the boundary of current region, move the window backward from the final position in steps of 1 pixels, the total length of this movement is calculated in the next statement. After each step, calculate the ccrrelation confidence between the two new corresponding positions, replace correlation confidence of those pixels which lie in the new position when correlation in this position is bigger than old one;

back: Calculate the number of pixels by which the window can be moved backwards keeping the boundaries of the target area and its corresponding area within the first and the second picture respectively; within the pictures. For backing up choose the minimum of this lenghth and 3;

stop: end.

References

Barnea, Daniel I. and Harvey F. Silverman [1972], "A Class of Algorithms for Fast Digital Image Registration", IEEE Transactions on Computers, VOL. C-21, NO.2, Feburary, 1972, pp. 179-186.

Ganapathy, Sundaram [1975], "Reconstruction of Scenes Containing Polyhedra from Stereo Pair of Views", Stanford Artificial Intelligience Project Memo No. 272.

Guzman, Adolfo [1968], "Computer Recognition of Three-Dimensional Objects in a Visual Scene", MIT project MAC MAC-TR-59.

Hannah, Marsha J. [1974], "Computer Matching of Areas in Stereo Images", Stanford Artificial Intelligence Project Memo No. 239.

Hart, Peter E. [1969], "Stereographic Perception of 3-Dimensional Scene", SRI Project 7642

Julesz, B. [1961], "Binocular Depth Perception and Pattern Recognition", Proceedings of the fourth London Symposium on Information Theory, pp. 21-224.

Julesz, B. [1963], "Towards the Automation of Binocular Depth Perception", Proceedings of IFIP Congress, 1962, North Holland Publishing Co. Amsterdam, pp.439-443.

Levine, M. D., A. O'Handley, and G. M. Yagi [1973], "Computer Determination of Depth Maps", Jet Propulsion Laboratory, Pasadena, Ca.

Nevatia, Ramakant [1976], "Depth Measurement by Motion Stereo", Computer Graphics and Image Processing 5, 203-214, 1976.

Quam, Lynn H. [1971], "Computer Comparison of Pictures", Stanford Artificial Intelligence Project Memo No. 144.

Thompson, Clark [1975], "Depth Perception in Stereo Computer Vision", Stanford Artificial Intelligence Project Memo No. 268.